UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Evaluation and model selection for unsupervised outlier detection and one-class classification

Henrique Oliveira Marques

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Henrique Oliveira Marques

Evaluation and model selection for unsupervised outlier detection and one-class classification

Thesis submitted to the Institute of Mathematics
and Computer Sciences – ICMC-USP – in
accordance with the requirements of the Computer
and Mathematical Sciences Graduate Program, for
the degree of Doctor in Science. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof. Dr. Ricardo José Gabrielli
Barreto Campello

USP – São Carlos
October 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

M357e Marques, Henrique Oliveira
Evaluation and model selection for unsupervised
outlier detection and one-class classification /
Henrique Oliveira Marques; orientador Ricardo José
Gabrielli Barreto Campello. -- São Carlos, 2019.
150 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2019.

1. outlier detection. 2. semi-supervised
learning. 3. unsupervised learning. 4. internal
evaluation. 5. model selection. I. Campello,
Ricardo José Gabrielli Barreto, orient. II. Título.

Henrique Oliveira Marques

Avaliação e seleção de modelos em detecção não supervisionada de outliers e classificação de classe única

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Ricardo José Gabrielli Barreto Campello

USP – São Carlos
Outubro de 2019

ACKNOWLEDGEMENTS

Agradeço a toda minha família, em especial aos meus pais, Abimael e Alaise, que mesmo de longe sempre me motivaram e me deram todo apoio e carinho para sempre continuar lutando.

Ao Prof. Ricardo Campello pela sua dedicação e comprometimento com o meu crescimento profissional (que já vêm desde os tempos de mestrado) e também pela sua compreensão nos momentos difíceis (que não foram poucos). Sob sua orientação, eu tive um crescimento que antes de ingressar na USP eu jamais imaginei que um dia eu fosse atingir.

Ao Prof. José Fernando Rodrigues Júnior por toda a sua gentileza e disponibilidade para me ajudar sempre que fosse preciso.

Ao Prof. Jörg Sander (UofA) por ter me recebido sob sua supervisão durante o um ano de doutorado sanduíche no Canadá. Esse período foi de extrema importância não só para o desenvolvimento desta pesquisa, mas principalmente para o meu crescimento pessoal e profissional.

Ao Prof. Arthur Zimek (SDU) por todas as suas contribuições relevantes no desenvolvimento desta pesquisa, muito do que está neste trabalho é também devido a sua colaboração.

A todos os membros do comitê avaliador por dedicar parte do seu tempo para avaliar e fazer suas devidas contribuições afim de melhorar a qualidade deste trabalho. Tenham certeza que cada uma das contribuições serão consideradas e faremos de tudo para que elas sejam refletidas na versão final deste trabalho.

A todos meus amigos que compartilharam parte desta jornada comigo e me proporcionaram vários momentos de diversão para que essa jornada fosse mais leve, em especial aos amigos dos laboratórios BioCom e LabES do ICMC-USP, do ENACTUS CAASO-USP e do Department of Computing Science e do Triathlon Club da University of Alberta.

A todos os funcionários diretos e indiretos da USP, em especial aos funcionários do restaurante universitário, limpeza, segurança, manutenção, CEFER e técnicos administrativos. Por muitas vezes vocês fazem todo o trabalho invisível sem que recebam o devido reconhecimento por isso, o trabalho de vocês torna a nossa rotina muito mais fácil.

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo suporte financeiro ao projeto, tanto na bolsa regular de doutorado (Processo FAPESP #2015/06019-0), quanto na Bolsa Estágio de Pesquisa no Exterior (BEPE) (Processo FAPESP #2017/04161-0) e a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo auxílio financeiro na fase inicial do projeto.

ON THE PLUMAGE OF BIRDS

Before the discovery of Australia, people in the Old World were convinced that all swans were white, an unassailable belief as it seemed completely confirmed by empirical evidence. The sighting of the first black swan might have been an interesting surprise for a few ornithologists (and others extremely concerned with the coloring of birds), but that is not where the significance of the story lies. It illustrates a severe limitation to our learning from observations or experience and the fragility of our knowledge. One single observation can invalidate a general statement derived from millennia of confirmatory sightings of millions of white swans. All you need is one single (and, I am told, quite ugly) black bird. [...]

First, it is an outlier, as it lies outside the realm of regular expectations, because nothing in the past can convincingly point to its possibility. Second, it carries an extreme impact. Third, in spite of its outlier status, human nature makes us concoct explanations for its occurrence after the fact, making it explainable and predictable.

Nassim Nicholas Taleb,

The Black Swan: The Impact of the Highly Improbable

ABSTRACT

MARQUES, H. O. **Evaluation and model selection for unsupervised outlier detection and one-class classification**. 2019. 150 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Outlier detection (or anomaly detection) plays an important role in the pattern discovery from data that can be considered exceptional in some sense. An important distinction is that between the supervised, semi-supervised and unsupervised techniques. In this work, we focus on semi-supervised and unsupervised techniques. It has been shown that unsupervised outlier detection techniques can be adapted to be applicable also in the semi-supervised setting. Therefore, we conduct a comparative study between the semi-supervised techniques and unsupervised techniques adapted to the semi-supervised context. The main focus of this work, however, is on the unsupervised *evaluation* of outlier detection. Although there is a large and growing literature that tackles the outlier detection problem, the unsupervised evaluation of outlier detection results is still virtually untouched in the literature, especially in the context of unsupervised detection. The so-called internal evaluation, based solely on the data and the assessed solutions themselves, is required if one wants to statistically validate (in absolute terms) or just compare (in relative terms) the solutions provided by different algorithms or by different parameterizations of a given algorithm in the absence of labeled data. However, in contrast to cluster analysis, where indexes for internal evaluation and validation of clustering solutions have been conceived and shown to be very useful, in the outlier detection domain this problem has been notably overlooked. Here we discuss this problem and provide solutions for the internal evaluation of outlier detection results. In the scenario of semi-supervised detection, we propose an (relative) internal evaluation measure based on data perturbation and compared it with the main measures of the literature, providing the reader with clear recommendations of the best scenario for the use of each one. In the scenario of unsupervised detection, the pioneering measure for internal evaluation of binary outlier solutions, proposed by the author of this thesis in his master's work, is extended to the more general scenario of non-binary outlier solutions, which involves the evaluation of outlier detection scorings, which is the type of result produced by most widely used database-oriented algorithms in the literature. We extensively evaluate both measures in several experiments involving different collections of synthetic and real datasets collected from public repositories.

Keywords: Outlier detection, semi-supervised learning, unsupervised learning, internal evaluation, model selection.

RESUMO

MARQUES, H. O. **Avaliação e seleção de modelos em detecção não supervisionada de outliers e classificação de classe única**. 2019. 150 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

A área de detecção de outliers (ou detecção de anomalias) possui um papel fundamental na descoberta de padrões em dados que podem ser considerados excepcionais sob alguma perspectiva. Uma importante distinção se dá entre as técnicas supervisionadas, semissupervisionadas e não supervisionadas de detecção. O presente trabalho enfoca as técnicas de detecção semissupervisionadas e não supervisionadas. As técnicas não supervisionadas de detecção podem ser adaptadas para operarem também de forma semissupervisionada. Desta forma, foi realizado um estudo comparativo entre as técnicas de detecção semissupervisionada e as técnicas não supervisionadas adaptadas ao contexto semissupervisionado. O principal foco deste trabalho, no entanto, está na *avaliação* não supervisionada de detecção de outliers. Embora exista uma literatura grande e crescente que aborde o problema de detecção de outliers, a avaliação não supervisionada dos resultados em detecção de outliers ainda está praticamente intocada na literatura, especialmente no contexto de detecção não supervisionada. A chamada avaliação interna, que baseia-se unicamente nos dados e nas próprias soluções a serem avaliadas, é necessária se for preciso validar estatisticamente (em termos absolutos) ou apenas comparar (em termos relativos) as soluções fornecidas por diferentes algoritmos ou por diferentes parametrizações de um dado algoritmo na ausência de dados rotulados. No entanto, em contraste com agrupamento de dados, onde os índices para validação e avaliação interna de soluções de agrupamento foram concebidos e demonstraram ser bastantes úteis, no domínio de detecção de outliers, este problema tem sido notavelmente negligenciado. Nesta tese, este problema é discutido e soluções são fornecidas para a avaliação interna dos resultados em detecção de outliers. No cenário de detecção semissupervisionada, uma medida (relativa) de avaliação interna baseada na perturbação dos dados é proposta e comparada com as principais medidas da literatura, fornecendo ao leitor recomendações claras do melhor cenário para a utilização de cada uma delas. No cenário de detecção não supervisionada, a medida pioneira para avaliação interna de soluções binárias de detecção de outliers, proposta pelo autor desta tese em seu trabalho de mestrado, é estendida para o cenário mais geral de soluções não binárias de detecção de outliers, que envolve a avaliação de scorings de detecção de outliers, que é o tipo de resultado produzido pela ampla maioria dos algoritmos. Ambas medidas são extensivamente avaliadas em vários experimentos envolvendo diferentes coleções de bases de dados sintéticas e reais coletadas de repositórios públicos.

Palavras-chave: Detecção de outliers, aprendizado semissupervisionado, aprendizado não supervisionado, avaliação interna, seleção de modelos.

LIST OF ABBREVIATIONS AND ACRONYMS

ε_i	Inlier rejection
ε_o	Outlier acceptance
m_{clSize}	Minimum cluster size
m_{cl}	Maximum clump size
ABOD	Angle-Based Outlier Detection
ABOF	Angle-Based Outlier Factor
CD	Critical Difference
CLT	Central Limit Theorem
COF	Connectivity-based Outlier Factor
EPD	Edge Pattern Detection
FN	False Negative
FP	False Positive
GLOSH	Global-Local Outlier Scores from Hierarchies
INFLO	Influenced Outlierness
IREOS	Internal, Relative Evaluation of Outlier Solutions
KDE	Kernel Density Estimation
KDEOS	Kernel Density Estimation Outlier Score
KLR	Kernel Logistic Regression
kNN	k -Nearest Neighbor
kNN_{global}	k -Nearest Neighbor
kNN_{local}	k -Nearest Neighbor Data Description
kNNDD	k -Nearest Neighbor Data Description
kNNW	k NN-weight
LDF	Local Density Factor
LDOF	Local Distance-based Outlier Factor
LOCI	Local Correlation Integral
LOF	Local Outlier Factor
LoOP	Local Outlier Probabilities
LP	Linear Programming
lrd	Local reachability density
MCC	Matthews Correlation Coefficient

MDEF	Multi-Granularity Deviation Factor
MST	Minimum Spanning Tree
OC-SVM	One-Class Support Vector Machine
ODIN	Outlier Detection using Indegree Number
PW	Parzen Window
QP	Quadratic Programming
RkNNs	Reverse k NNs
ROC AUC	Area Under the ROC curve
SDS	Self-adaptive Data Shifting
SVDD	Support Vector Data Description
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

CONTENTS

1	INTRODUCTION	17
1.1	Research Hypotheses	20
1.2	Outline of the Thesis	21
2	OUTLIER DETECTION	23
2.1	Supervised Outlier Detection	23
2.2	Semi-supervised Outlier Detection	24
2.2.1	<i>Density Methods</i>	25
2.2.2	<i>Boundary Methods</i>	27
2.2.3	<i>Reconstruction Methods</i>	31
2.3	Unsupervised Outlier Detection	31
2.3.1	<i>Statistical Methods</i>	32
2.3.2	<i>Distance Methods</i>	33
2.3.3	<i>Density Methods</i>	34
2.3.4	<i>Angle Methods</i>	39
2.4	Chapter Remarks	40
3	ON THE EVALUATION OF OUTLIER DETECTION AND ONE- CLASS CLASSIFICATION METHODS	41
3.1	Introduction	42
3.2	Adapting Unsupervised Outlier Detection Methods to the One- Class Classification Setting	43
3.2.1	<i>Adapting LOF and its Variants to One-Class Classification</i>	45
3.2.2	<i>Adapting GLOSH to One-Class Classification</i>	46
3.3	Experimental Setup	48
3.3.1	<i>Datasets</i>	48
3.3.2	<i>Labeling of Datasets</i>	49
3.3.3	<i>Evaluation Measures</i>	50
3.3.4	<i>Performance Evaluation</i>	53
3.4	Results	54
3.5	Chapter Remarks	59
4	MODEL SELECTION FOR ONE-CLASS CLASSIFICATION	61
4.1	Introduction	61

4.2	Model Selection Methods for One-Class Classification	63
4.2.1	<i>Uniform Object Generation</i>	64
4.2.2	<i>Self-Adaptive Data Shifting</i>	64
4.2.3	<i>Data Perturbation</i>	64
4.2.4	<i>Complexity</i>	67
4.3	Experiments	68
4.3.1	<i>Results</i>	69
4.4	Chapter Remarks	73
5	INTERNAL EVALUATION OF UNSUPERVISED OUTLIER DE- TECTION	75
5.1	Introduction	75
5.2	Related Work	78
5.3	Internal Evaluation of Outlier Detection	79
5.3.1	<i>Problem Statement</i>	79
5.3.2	<i>Preliminary Attempt: A Baseline Index</i>	80
5.3.3	<i>IREOS Index</i>	85
5.3.4	<i>Adjustment for Chance</i>	89
5.3.5	<i>Complexity</i>	95
5.4	Evaluation	96
5.4.1	<i>Datasets</i>	96
5.4.2	<i>Methods and Measures</i>	98
5.4.3	<i>Results</i>	101
5.5	Chapter Remarks	105
6	SPEEDUP TECHNIQUES FOR IREOS	107
6.1	Separability Curves	107
6.2	Separability Values Across the Dataset	111
6.3	Classifiers and Computational Complexity Revisited	112
6.4	Evaluation	115
6.4.1	<i>Efficiency of Approximations</i>	115
6.5	Chapter Remarks	120
7	CONCLUSIONS	121
7.1	Future Works and Limitations	122
	BIBLIOGRAPHY	125
APPENDIX A	TABLES OF RESULTS, CHAPTER 3	135
APPENDIX B	TABLES OF RESULTS, CHAPTER 4	143

INTRODUCTION

The information technology has evolved extraordinarily. The computer processing capacity has increased over the past 65 years following Moore's law, which states that the processing capacity increases by approximately 100% every 18 months (MOORE, 1964; MOORE, 1965; MOORE, 1975). At the same time, our ability to generate and collect data from a variety of sources (*e. g.*, satellites, sensor networks, telescopes, surveillance systems, etc) has grown even faster and at lower costs. The widespread use of the internet, bar codes, RFID and computer transactions, for example, has led to very large databases in many areas of human activity. Paradoxically, these technological developments have caused a burst of data, such that our ability to collect and store data has far surpassed our ability to manually analyze and extract knowledge from them (FAYYAD *et al.*, 1996). Due to our inability to manually analyze this huge amount of data, researchers from a variety of fields have been engaged in the development of techniques capable of synthesizing, processing and transforming data into useful knowledge in an intelligent and automated way, for what has been called data mining.

The term data mining (FAYYAD *et al.*, 1996; HAN; KAMBER, 2006; WITTEN; FRANK; HALL, 2011) widely refers to statistical, mathematical and computational techniques of analysis to effectively and efficiently extract useful knowledge from large datasets. There are some central tasks in data mining that together cover most of the existing problems in practical applications (TAN; STEINBACH; KUMAR, 2005): association analysis, cluster analysis, pattern classification, regression analysis, and outlier detection. Among such tasks, the area of outlier (anomaly) detection plays an important role in discovering patterns that are exceptional in some sense (HODGE; AUSTIN, 2004; PATCHA; PARK, 2007; HADI; IMON; WERNER, 2009; CHANDOLA; BANERJEE; KUMAR, 2009; CHANDOLA; BANERJEE; KUMAR, 2012; ZIMEK; SCHUBERT; KRIEGEL, 2012). Detecting such patterns is relevant for two main reasons: (i) in some applications, such patterns represent spurious data (*e.g.*, sensor failures or noise) that should be removed in a preprocessing step for further data analysis; or, more importantly, (ii) in many applications, such patterns represent extraordinary behaviors that deserve some special

attention, such as genes associated with certain diseases, frauds in financial systems, employees with unusual productivity profiles, or customers with uncommon purchasing behaviors. In this context, it has been said that “*one person’s noise is another person’s signal*” (KNORR; NG, 1998; HAN; KAMBER, 2006).

Outlier detection techniques can be categorized in different ways. For instance, a common distinction is that between supervised, semi-supervised and unsupervised outlier detection techniques (TAN; STEINBACH; KUMAR, 2005). Supervised techniques assume that a set of observed instances labeled as inliers and outliers are available to train a classifier. In the semi-supervised scenario, labeled outliers are usually not available and only previously known inliers can be used in order to obtain a (one class) classification model. When no labeled data are available at all, it is necessary to use unsupervised techniques, which do not assume any prior knowledge about which instances are outliers and which are inliers.

In this work, we focus on the semi-supervised and unsupervised outlier detection scenarios. In the semi-supervised scenario, outliers may be defined as instances that differ, in some sense, from the data available for training. In this context, also referred to as novelty detection (PIMENTEL *et al.*, 2014), one-class classification models are typically used to fit the training data and new instances that do not fit in the model are deemed outliers. In the unsupervised scenario, the notion of “outlierness” is not precisely and generally defined, in fact, it remains rather vague, such as these classic definitions:

“An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs” (GRUBBS, 1969).

“An observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism” (HAWKINS, 1980).

“An observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data” (BARNETT; LEWIS, 1994).

Outlier detection algorithms aim to automatically identify instances with such characteristics in data sets. In order to solve this task, a rich variety of methods has been developed, from classic parametric statistical methods (HAWKINS, 1980; BARNETT; LEWIS, 1994; MARKOU; SINGH, 2003) to more recent database-oriented approaches conceived to deal with multivariate, possibly large databases. Each of these algorithms, however, uses its own criterion to judge quantitatively the level of adherence of each instance with the concept of outlier, from a particular perspective. This complicates not only the selection of a particular algorithm and/or the choice of an appropriate configuration of parameters for this algorithm in a practical application, but also the assessment of the *quality* of the solutions obtained, especially in light of the problem of

defining a measure of quality that is not tied to the criteria used by the algorithms themselves. These issues are interrelated and refer to the problems of model selection and assessment (evaluation or validation) of results in unsupervised learning. These problems have been investigated for decades in the area of data clustering (JAIN; DUBES, 1988), but are rarely mentioned and are virtually untouched in the area of outlier detection (ZIMEK; CAMPELLO; SANDER, 2013).

In the literature, the procedure for model selection and evaluation of results in outlier detection usually is based on the use of previously labeled datasets, in which the outliers (according to a particular intuition or semantic) are previously known. In this scenario, referred to as external evaluation or validation, the labels are not used by the algorithms, but rather to assess their results only (ZIMEK *et al.*, 2013; ZIMEK; SCHUBERT; KRIEGEL, 2012). Specifically, the labeling or ranking produced by the algorithm is compared with the correct and previously known labeling (ground truth). The quantitative result of this comparison is taken as the quality measure of the outlier detection algorithm using, for example, precision-at- n or ROC AUC curve (SCHUBERT *et al.*, 2012; CAMPOS *et al.*, 2016). However, the availability of a fully labeled dataset required by external evaluation measures is not consistent with the premises of semi-supervised and unsupervised learning, and the commonly practiced external evaluation of semi-supervised and unsupervised outlier detection algorithms makes sense only when comparing performances of algorithms in controlled experiments. Therefore, although the evaluation procedure described above is useful during the development of new algorithms, it is feasible in practical applications only in the supervised context.

In the data clustering domain, the related problems of evaluation and model selection are tackled by using some kind of quantitative index, called validation criterion (JAIN; DUBES, 1988). In practice, when labels are not available, *internal* validation indexes can be used. These indexes are called internal as they do not make use of any external information (such as class labels) in the evaluation of a solution. Instead, internal indexes measure the quality of an obtained clustering solution based only on the solution and the data instances. Most such indexes are also *relative* in the sense that they can be employed to compare different clustering solutions pointing out which one is better in relative terms. Therefore they can also be used for model selection. Internal, relative indexes have been shown to be effective and useful tools for the unsupervised clustering evaluation and model selection tasks — e.g. see (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001; MILLIGAN; COOPER, 1985; VENDRAMIN; CAMPELLO; HRUSCHKA, 2010; VENDRAMIN; JASKOWIAK; CAMPELLO, 2013) and references therein.

In this work, we discuss and provide solutions for evaluation and model selection of unsupervised outlier detection and one-class classification. In the scenario of one-class classification, we propose an (relative) internal index based on data perturbation and compared it with the few existing in the literature. In the scenario of unsupervised outlier detection, the pioneering measure for internal evaluation of binary outlier solutions, proposed by the author of this thesis during his master’s work (MARQUES *et al.*, 2015; MARQUES, 2015), is extended to

the more general scenario of non-binary outlier solutions, which is the type of result produced by most widely used database-oriented algorithms in the literature. Although the focus of this work is on the evaluation and model selection of unsupervised outlier detection and one-class classification, the adaptation of unsupervised methods to operate in the semi-supervised setting is also one of the research hypotheses investigated in this work. In the following, we present all the research hypotheses investigated in this work.

1.1 Research Hypotheses

Considering the existing problems in the literature of semi-supervised and unsupervised outlier detection, in particular, those related to the evaluation and model selection, this work investigated the following four main hypotheses:

- **Hypothesis I:** The performance of unsupervised outlier detection algorithms adapted to the one-class classification problem has been previously evaluated in the literature only in the scenario of unimodal inlier class. In application scenarios other than the one evaluated in the literature, *e.g.*, in the scenario of multi-modal inlier class, the performance of these algorithms is different.
- **Hypothesis II:** Given the existence of only a few methods for model selection in the literature of one-class classification, a new model selection method for one-class classification can be developed that improves on the existing methods in general and specific application scenarios.
- **Hypothesis III:** IREOS ([MARQUES *et al.*, 2015](#); [MARQUES, 2015](#)) was originally designed to evaluate and compare different binary candidate outlier detection solutions. Therefore, IREOS can be extended to evaluate and compare the more general scenario of non-binary outlier solutions, which is the type of result produced by most widely used database-oriented algorithms in the literature.
- **Hypothesis IV:** The extended version of IREOS can no longer be computed efficiently by taking advantage of the imbalanced nature of the binary outlier detection problem. The computational burden can be mitigated by speedup techniques to approximate the exact index.

1.2 Outline of the Thesis

The remainder of this thesis is organized as follows:

Chapter 2 introduces the basic concepts of the algorithms for supervised, semi-supervised, and unsupervised outlier detection. Since the focus of this thesis is on the semi-supervised and unsupervised algorithms, the chapter gives more emphasis on them and just briefly overviews the main approaches for supervised scenarios.

Chapter 3 focus on the comparison of one-class classification algorithms with unsupervised outlier detection algorithms adapted to one-class classification. The chapter discusses the adaptation of such algorithms, and compares, in a rigorous experimental setup, a number of one-class classification and unsupervised outlier detection algorithms using a large number of datasets, and different performance measures.

Chapter 4 discusses the problem of model selection in one-class classification. A new method is proposed and compared with the main model selection methods of the one-class classification literature, in the same rigorous experimental setup as chapter 3.

Chapter 5 discusses the problem of internal evaluation of unsupervised outlier detection results. In this chapter, the index for binary solutions called IREOS is described, and extended to the more general case of non-binary solutions. An extensive evaluation involving different collections of synthetic and real datasets is performed to evaluate the extended version of the index.

Chapter 6 introduces many different ways to mitigate the computational burden of the extended version of IREOS. The speed up strategies are evaluated in experiments to assess the trade-off between computational efficiency and efficacy.

Finally, Chapter 7 summarizes the main topics discussed in the previous chapters and concludes the thesis. It also points out some limitations and opportunities for future work.

OUTLIER DETECTION

Outlier detection (or anomaly detection) plays an important role in the pattern discovery from data that can be considered exceptional in some sense. Detecting such patterns is relevant because, in many applications, these patterns represent extraordinary behaviors that deserve some special attention, such as genes associated with certain diseases and frauds in financial systems.

In this chapter, we introduce the basic concepts of outlier detection. An important distinction is that between the supervised, semi-supervised, and unsupervised techniques. In section 2.1, we provide a very brief overview of the supervised outlier detection. We present, in more details, the semi-supervised and unsupervised techniques, respectively, in sections 2.2 and 2.3. Our goal is to provide the reader with the assumptions behind the algorithms used in the following chapters, especially, the semi-supervised and unsupervised algorithms, which are the focus of this thesis.

2.1 Supervised Outlier Detection

The supervised techniques assume that a set of observed instances labeled as inliers and outliers are available to train a classifier. The supervised outlier detection problem may be considered a special case of the binary classification problem. The classification problem has been extensively studied, and an extensive amount of algorithms are available in the literature (DUDA; HART; STORK, 2000; BISHOP, 2006). However, due to the rare nature of anomalies, it is natural for the distribution of the classes to be very imbalanced, which makes the problem more challenging to create robust and generalized models. Moreover, outliers are not always easily separated from inliers. In some applications, the outliers try to simulate the behavior of an inlier (*e.g.*, in frauds), making the problem even more challenging.

The classification problem has also been widely studied in the context of class imbalance

(BATISTA; PRATI; MONARD, 2004) and adapted for use in the supervised outlier detection (AGGARWAL, 2013). The straightforward use of traditional evaluation metrics and classifiers may lead to very misleading results due to the imbalance problem. For example, consider a case where 99% of the instances are normal, and only the remaining 1% is abnormal. A trivial classification algorithm, in which every instance is labeled as normal, would have a very high accuracy of 99%, even though it would not have classified a single abnormal instance correctly. In outlier detection, this problem is quite critical, since the abnormal class is the class of interest and the misclassification of anomalous instances is a problem much more serious than the misclassification of normal instances. For example, diagnosing a cancer patient as healthy (which could lead to the death of the patient due to the lack of treatment) is much more serious than diagnosing a healthy patient as cancerous (which would probably be fixed with posterior exams). There are two common approaches applied in this context to deal with class-imbalanced:

- **Cost Sensitive Learning:** The cost sensitive learning tackles the problem at the algorithmic level by introducing weights associated with the instances. These weights correspond to misclassification cost, and they are incorporated into the objective function of the classifier. In order to adjust the class distribution, the class with fewer instances should receive higher weights. The objective function of most classification algorithm requires only minor changes to introduce these weights (TING, 2002; TANG *et al.*, 2009).
- **Sampling:** In contrast to the previous approach, the sampling methods tackle the imbalance problem at the data level. The goal of the sampling methods consists of balancing classes by re-sampling the original dataset. Either the rare class can be oversampled (CHAWLA *et al.*, 2002), or the normal class can be under-sampled (KUBAT; MATWIN, 1997), or both samplings can be combined (BATISTA; PRATI; MONARD, 2004).

Although supervised techniques have such challenges, they assume an appropriate set of labeled instances. Such a set can be used for training and validation of the model using well established classification techniques, such as hold-out, cross-validation, and bootstrap. For the model evaluation, the common approach is to use external evaluation, in which the labeling or ranking produced by the algorithm is compared with the correct and previously known labeling (ground truth). The quantitative result of this comparison is taken as the quality measure using, for example, precision-at- n or ROC AUC (SCHUBERT *et al.*, 2012; CAMPOS *et al.*, 2016).

2.2 Semi-supervised Outlier Detection

Unlike the supervised outlier detection, in the semi-supervised outlier detection, it is usually assumed that we are only provided with previously known inliers. The outliers are either nonexistent or very rare and insufficient for the generalization of a classifier. In this scenario,

also referred to as novelty detection (PIMENTEL *et al.*, 2014), the model is typically obtained using one-class classification models (TAX, 2001). Using only the previously known inliers, these models can generalize the inlier class, such that any other instance that does not belong to the inlier class is rejected. In this context, the instances not belonging to the inlier class are called outliers.

Several models have been proposed to solve the one-class classification problem. We can categorize such models into density methods, boundary methods, and reconstruction methods. For each of these categories, there are a number of algorithms. In the following sections, we present and discuss these three categories, as well as some examples of algorithms.

2.2.1 Density Methods

The density methods estimate the probability density function of the inlier class, given by the training data, and threshold this density. This threshold can be obtained either analytically from the data distribution or empirically from the training set. The new instances are classified using the threshold of the estimated function. The drawback, however, is that estimating densities is not an easy problem. There are two approaches to estimate densities:

- **Parametric:** The parametric approaches assume a parametric data distribution, such as Gaussian or Poisson distribution, and discordancy tests are used to test new instances (HAWKINS, 1980; BARNETT; LEWIS, 1994). The disadvantage of this approach is that data distribution is generally not known a priori, and assuming a particular distribution for the data imposes a very restrictive model, which results in a large bias when the assumption is violated.
- **Non-parametric:** The non-parametric models do not assume any distribution, which results in very flexible models. The disadvantage, however, is that it requires a sufficiently large sample of inliers to produce a good estimation. Depending, *e.g.*, on the dimensionality of the problem, the number of instances required to sufficiently represent the underlying distribution can become very large, and may prove too expensive to obtain in a real-world setting.

Despite the difficulties in estimating densities, the density methods often outperform the others when the assumptions are satisfied (see chapter 3). In the following, we introduce an example of a parametric and a non-parametric algorithm.

Gaussian Density

The simplest and most widely used parametric model is the Gaussian distribution (BISHOP, 2006). In the Gaussian model, the Gaussian probability density function:

$$p_{Gauss}(\mathbf{x}|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)} \quad (2.1)$$

is fit to the inlier data, where μ is the mean and Σ is the covariance matrix, and d is the dimensionality of the data. A new instance can be classified by computing its probability under the learned distribution.

Note that the quantity

$$\text{Mahalanobis}(\mathbf{x}) = \sqrt{(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)} \quad (2.2)$$

which appears in the exponent in Equation (2.1) is the Mahalanobis distance (MAHALANOBIS, 1936). The use of the squared Mahalanobis distance, instead of the probability density function, to classify new instances avoids underflows and unnecessary calculations. An important advantage of the normal density is that, when we assume that the data is normally distributed, the distribution of the Mahalanobis distance follows a χ_d^2 distribution with d degrees of freedom. Therefore, we can compute analytically the optimal threshold.

Aside from the inherited disadvantages of parametric methods, the main disadvantages of the Gaussian density are related to the covariance matrix. For high dimensional data, this model suffers from very large covariance matrices, which makes expensive to compute its inverse. In case the inverse of the covariance matrix cannot be calculated (*e. g.* data with singular directions), it has to be approximated, for example, by adding a small constant ε to the diagonal. In that case, the user has to supply the regularization parameter ε . Another alternative is to approximate it by the pseudo-inverse (STRANG, 2016).

PW

Parzen Window (PW) is a non-parametric method based on Parzen Density Estimation (PARZEN, 1962) which estimates the density of the data using a mixture of kernels centered on each of the N individual training instances. The most often used kernel is the Gaussian kernels with diagonal covariance matrices $\Sigma_i = \Lambda I$, where Λ is a width parameter. The probability of an instance being an inlier is then computed as:

$$\text{PW}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N p_{Gauss}(\mathbf{x}|\mathbf{x}_i, \Lambda I) \quad (2.3)$$

In contrast to the Gaussian density, the threshold cannot be computed analytically, and we have to set a threshold empirically based on the training data. The training for the Parzen

density consists of the determination of the parameter Λ , which can be optimized using the maximum likelihood solution (DUIN, 1976). Alternatively, the user can supply the parameter Λ . In the latter, the computational cost for training is almost null. Testing new instances, however, is expensive. During the testing phase, the distances of the new instance to the training instances have to be computed, which also imposes a storage limitation, since the training instances have to be stored. This is main limitation of the method, especially because large training sets are required to produce a good density estimation.

2.2.2 Boundary Methods

Similarly to the non-parametric density methods, the boundary methods do not make any assumption about the data distribution, but instead of using a large number of samples to estimate the class density, they propose to solve the problem more simply. The boundary methods define a boundary around the training data, such that new instances that fall within the boundary are classified as inliers, while instances falling outside of the boundary are classified as outliers. Since we are only interested in defining this boundary, it is not necessary to obtain a large number of samples to fully represent the inlier class. Another advantage by defining this boundary is that the threshold to classify new instances is obtained directly. On the other hand, their focus on the boundary often causes the decision boundary to be affected by outliers/noises that might occur in the training set. Another disadvantage is that, due to the boundary methods rely on the distances between instances, they tend to be sensitive to the scaling of the features.

Due to their simplicity, the boundary methods are quite popular. However, such methods can be very costly due to the optimization problem that they have to solve. In the following, we introduce four boundary algorithms: One-Class Support Vector Machine (OC-SVM), Support Vector Data Description (SVDD), Linear Programming (LP), and k -Nearest Neighbor Data Description (kNNDD).

OC-SVM

The One-Class Support Vector Machine (OC-SVM) (SCHÖLKOPF *et al.*, 2001) is a boundary-based one-class classification method inspired by Support Vector Machines (SVM) (VAPNIK, 1995) used in regular classification problems. The primary difference between SVM and OC-SVM is that while SVM attempts to separate two or more classes with a maximum margin hyperplane, OC-SVM tries to find the optimal linear hyperplane that separates the dataset from the origin with maximal margin. It results in an optimization problem that is very similar to

the SVM:

$$\begin{aligned} \min_{\mathbf{w}, \rho, \xi} \quad & \frac{1}{2} \|\mathbf{w}\|^2 - \rho + \frac{1}{N\nu} \sum_i^N \xi_i \\ \text{subject to} \quad & \mathbf{w} \cdot \mathbf{x}_i \geq \rho - \xi_i, \quad \xi_i \geq 0 \end{aligned} \quad (2.4)$$

where $\mathbf{w}$ is the hyperplane which separates the instances $\mathbf{x}_i$ from the origin with margin ρ . ξ_i are slack variables allowing training instances $\mathbf{x}_i$ to fall outside the OC-SVM boundary, and ν is a penalty (regularization) parameter.

Note that just as traditional SVM cannot handle nonlinearly separable problems in the input space, OC-SVM cannot always separate the dataset from the origin in the input space. Therefore, it may be necessary to use kernel functions to transform the input space. The use of kernels such as the Gaussian kernel where $K(\mathbf{x}_i, \mathbf{x}_i) = 1, \forall_i$ ensures that any dataset is separable from the origin after it is mapped into feature space (SCHÖLKOPF *et al.*, 2001). The freedom to use different types of kernel functions makes the simple hyperplane in the feature space corresponds to a variety of nonlinear estimators in input space, such as Gaussian in the case of the Gaussian kernel.

SVDD

The Support Vector Data Description (SVDD) (TAX; DUIN, 2004) is another boundary method inspired by SVM. Unlike the SVM and OC-SVM that find the optimal linear hyperplane to maximize the margins, the SVDD encloses the inlier class in a minimum volume hypersphere by solving the following optimization problem:

$$\begin{aligned} \min_{R, \xi} \quad & R^2 + C \sum_i^N \xi_i \\ \text{subject to} \quad & \|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 + \xi_i, \quad \xi_i \geq 0 \end{aligned} \quad (2.5)$$

where R is the radius of the hypersphere, $\mathbf{a}$ is the center of the hypersphere, ξ_i are slack variables, and C is a penalty (regularization) parameter.

When data in the original SVDD and OC-SVM formulation are transformed to unit norm vectors $\left(\mathbf{x}' = \frac{\mathbf{x}}{\|\mathbf{x}\|}\right)$, the equivalence between the SVDD and the OC-SVM can be shown (TAX, 2001). By substituting the unit vector in Equation (2.5), we obtain the following optimization problem:

$$\begin{aligned} \min_{R', \xi'} \quad & R'^2 + C' \sum_i^N \xi'_i \\ \text{subject to} \quad & \|\mathbf{x}'_i - \mathbf{a}'\|^2 \leq R'^2 + \xi'_i, \quad \xi'_i \geq 0 \end{aligned} \quad (2.6)$$

where $\mathbf{x}'$ and $\mathbf{a}'$ are normalized vectors. Rewriting the constraint gives:

$$\begin{aligned} \|\mathbf{x}_i\|^2 - 2\mathbf{a}' \cdot \mathbf{x}'_i + \|\mathbf{a}'\|^2 &\leq R'^2 + \xi'_i \\ 1 - 2\mathbf{a}' \cdot \mathbf{x}'_i + 1 &\leq R'^2 + \xi'_i \\ \mathbf{a}' \cdot \mathbf{x}'_i &\geq \frac{1}{2}(2 - R'^2 - \xi'_i) \end{aligned} \quad (2.7)$$

By introducing an extra constant -2 and rewriting the constraint, we get:

$$\begin{aligned} \min_{R', \xi'} \quad & R'^2 - 2 + C' \sum_i^N \xi'_i \\ \text{subject to} \quad & \mathbf{a}' \cdot \mathbf{x}'_i \geq \frac{1}{2}(2 - R'^2 - \xi'_i), \quad \xi'_i \geq 0 \end{aligned} \quad (2.8)$$

We redefine:

$$\mathbf{w} = \mathbf{a}', \rho = \frac{1}{2}(2 - R'^2), \frac{1}{Nv} = C', \xi_i = \frac{1}{2}\xi'_i \quad (2.9)$$

and we have the following optimization problem:

$$\begin{aligned} \min_{\rho, \xi} \quad & 2 \left(-\rho + \frac{1}{Nv} \sum_i^N \xi_i \right) \\ \text{subject to} \quad & \mathbf{w} \cdot \mathbf{x}_i \geq \rho - \xi_i, \quad \xi_i \geq 0 \end{aligned} \quad (2.10)$$

which differs from OC-SVM formulation (Equation (2.4)) only by the factor 2 and $\frac{1}{2}\|\mathbf{w}\|^2$. However, since the vector $\mathbf{w}$ is normalized, $\|\mathbf{w}\|^2 = 1$ is also a constant. Therefore, when rescaled to unit norm both methods learn the same decision boundary.

Like traditional SVMs, the OC-SVM and SVDD formulation can also be extended to non-linearly transformed spaces using kernel methods. In the case of a kernel with the property $K(\mathbf{x}_i, \mathbf{x}_i) = 1, \forall_i$, such as the Gaussian kernel, the data is implicitly rescaled to unit norm (TAX, 2001). Therefore, the solutions of the SVDD and the OC-SVM are identical when such kernels, with equal parameters (for example, width for Gaussian kernel), and $C = \frac{1}{Nv}$ are used.

LP

The Linear Programming (LP) (PEKALSKA; TAX; DUIN, 2002) is a boundary method which, instead of using the explicit feature space, utilizes a dissimilarity measure to compare new instances to the inlier instances in the training set. The basic assumption is that instances belonging to the inlier class are similar to each other, while the outliers are dissimilar. The use of a dissimilarity measure makes the method very convenient for applications where there is a lack of good or suitable features due to the difficulty of defining them, as *e.g.* in unstructured data (strings, graphs or shapes). The dissimilarity measure does not need to be metric. However, it must meet the following criteria: reflectivity ($\mathcal{D}(x, x) = 0$), positivity ($\mathcal{D}(x, y) > 0$), and symmetry

($\mathcal{D}(x, y) = \mathcal{D}(y, x)$). Note that, in the dissimilarity space, the outliers are not close to the origin, but far away. Therefore, unlike the OC-SVM which find the hyperplane that separates the dataset from the origin with maximal margin, LP has to find a decision boundary as close as possible to the origin. LP constructs this boundary by minimizing the volume of a simplex (BAZARAA; JARVIS; SHERALI, 2009) with the main vertex being the origin and the other vertices resulting from the intersection of the boundary and the axes of the dissimilarity space. It results in the following formulation with the introduction of the slack variables:

$$\begin{aligned} \min_{\rho, \xi} \quad & \rho + \frac{1}{Nv} \sum_i^N \xi_i \\ \text{subject to} \quad & \mathbf{w} \cdot \mathcal{D}(x, y) \leq \rho + \xi_i, \quad \xi_i \geq 0, \\ & \sum w_j = 1, w_j \geq 0, \quad \rho \geq 0 \end{aligned} \tag{2.11}$$

which resembles the OC-SVM formulation.

For an unbounded $\mathcal{D}$, instances with large dissimilarities values (noises/outliers) may affect the decision boundary. In order to make the classifier robust against such values, $\mathcal{D}$ is transformed by the sigmoid function, $\text{sigm}(x) = \frac{2}{1+e^{x/s}}$, such that large values are bounded to 1. For the sigmoid, the user has to define the parameter s for its slope.

kNNDD

k -Nearest Neighbor Data Description (TAX, 2001) (kNNDD) avoids the explicit density estimation by only using the distances to the nearest neighbors. It resembles the local outlier detection methods (to be discussed in section 2.3) in that it approximates the local distance of the training instances, however in a simpler way. An instance is classified under kNNDD by computing the ratio between the distance from the instance to its k^{th} nearest neighbor $\text{NN}_k(\mathbf{x}_i)$, and the distance between the k^{th} nearest neighbor and its k^{th} nearest neighbor:

$$\text{kNNDD}_k(\mathbf{x}_i) = \frac{k\text{-dist}(\mathbf{x}_i)}{k\text{-dist}(\text{NN}_k(\mathbf{x}_i))} \tag{2.12}$$

where $k\text{-dist}(\mathbf{x}_i)$ denotes the distance from $\mathbf{x}_i$ to its k^{th} nearest neighbor.

Originally, the instance is classified as outlier when the $k\text{-dist}(\mathbf{x}_i)$ is larger than the $k\text{-dist}(\text{NN}_k(\mathbf{x}_i))$, i.e., $\text{kNNDD}_k(\mathbf{x}_i) > 1$. However, using such a threshold makes part of training instances to be classified as outliers. Therefore, a higher threshold can be used. The kNNDD has the same computational issues to test new instances than PW. However, according to the authors, the method performs best in small sample size, where computational efforts are low. For larger sample sizes and noisy data, the quality of the results is very poor (TAX, 2001).

2.2.3 Reconstruction Methods

The reconstruction methods provide a compact representation of the training data while attempting to preserving most of the information. In this representation of the data, the noise contribution is decreased, which makes these methods naturally more robust to noise, in contrast to the boundary methods. Once the model has been obtained, new instances can then be described through this model. The reconstruction error, the difference between the original instance and the described instance, indicates the resemblance of a new instance to the original training distribution. In order to classify new instances, the reconstruction error is used to characterize how well the instance fits the model. The outliers should be described worse than true inliers and their reconstruction error should be high. The threshold for the reconstruction error has to be obtained empirically using the training set.

Below, we introduce the auto-encoder networks as an example of reconstruction method.

Auto-Encoder Networks

In the Auto-Encoder networks ([JAPKOWICZ; MYERS; GLUCK, 1995](#)), a neural network with a sigmoidal transfer function, a single hidden layer, and a parameter-defined number of hidden units smaller than the number of input units is trained on the inlier class to learn a representation of the data. The network is trained to reproduce the input patterns at its output layer (*i.e.*, it should perform the identity operation) by minimizing the mean square error. In order to classify a new instance, we supply the instance as input to the network, and the difference between the original input and the network's output defines the reconstruction error.

The auto-encoder can learn very flexible models. However, it suffers from the same problems as the conventional neural networks for classification problems ([FREEMAN; SKAPURA, 1991](#)).

2.3 Unsupervised Outlier Detection

When no labeled data are available at all, it is necessary to use unsupervised techniques, which do not assume any prior knowledge about which instances are outliers and which are inliers. In unsupervised outlier detection, we are provided with a set of unlabeled instances and are tasked with determining whether each instance is an inlier or an outlier. In this context, an outlier may be defined as an instance “which appears to be inconsistent with the remainder of that set of data” ([BARNETT; LEWIS, 1994](#)). In the unsupervised outlier detection, however, there is no generally applicable definition of “appearance of inconsistency”; its formalization rather depends on the application scenario and the detection method to be used.

Since the seminal work of [Knorr and Ng \(1998\)](#), a plethora of unsupervised outlier detection algorithms has emerged. Each of these algorithms uses its own criterion to judge quantitatively the level of adherence of each instance with the concept of outlier, from a particular perspective. In a more general way, these algorithms can be categorized into “local” versus “global”. While the global outliers are unusual w.r.t. the complete dataset, the local outliers are unusual w.r.t. a reference set, such as their k NNs. For example, in Figure 1, the instance x can be considered a local outlier, as it is unusual w.r.t. the instances of the cluster c_1 . A global algorithm, however, could hardly detect it as an outlier, as it deviates, in absolute values, less than most instances in cluster c_2 deviate from their k NNs.

Figure 1 – Example dataset with a local outlier.

In a more specific way, we can categorize the unsupervised outlier detection algorithms according to the approach used to define whether an instance is outlier or not (or to which degree it is considered outlier). There are a variety of approaches, including statistical, distance, density, and angle-based.

2.3.1 Statistical Methods

The field of statistics emerged initially as the main research area for unsupervised outlier detection. Statistical approaches for unsupervised outlier detection usually assume that the inliers were generated by some known parametric distribution and outliers deviate from the underlying distribution. Numerous statistical/discordance tests have been proposed to measure such deviation ([HAWKINS, 1980](#); [BARNETT; LEWIS, 1994](#)). These tests have in common the fact that they need to make several assumptions. For example, the tests may assume the underlying distribution of the data, the number of expected outliers, the types of expected outliers (*e.g.*, upper or lower outliers), etc. In contrast to the parametric density methods (discussed in section 2.2.1), in the unsupervised setting, one must deal with possible outliers while estimating

model parameters. Such parameters (*e.g.*, mean and standard deviation) are rather sensitive to the presence of outliers. The parameter distortions generate undesirable effects known as masking, when outliers mask their own presence (resulting in false negatives), and swamping, when inliers are incorrectly labeled as outliers due to these distortions (resulting in false positives). In addition to these tests being parametric, most of them apply only to numeric and univariate datasets (KNORR; NG; TUCAKOV, 2000). For example, for one-dimensional Normal distribution, a common rule of thumb is to consider outliers instances that deviate more than three times the standard deviation from the mean; in the multidimensional case, an equivalent threshold can be set at Mahalanobis distance (discussed in section 2.2.1). In the context of unsupervised outlier detection, variants of the Mahalanobis distance were proposed using robust estimators to deal with the presence of outliers (ROUSSEEUV; ZOMEREN, 1990; HADI, 1992).

2.3.2 Distance Methods

Distance-based techniques assume that outliers are instances that are far away from their nearest neighbors. Therefore, it is necessary to define a distance or similarity measure between two instances. There are different distances and similarity measures. While for continuous attributes the Euclidean distance is the most usual, for categorical attributes is commonly used the simple matching coefficient (TAN; STEINBACH; KUMAR, 2005). Well-Known algorithms that use this approach are kNN, kNNW, LDOF, and ODIN, presented below.

kNN

The *k*-Nearest Neighbor (kNN) is global outlier detection algorithm proposed by Ramaswamy, Rastogi and Shim (2000). It is one of the simplest and most well-known distance-based algorithm existing. kNN assigns to each instance in the dataset a score reflecting its characterization as an outlier. In order to compute the score of a given instance $\mathbf{x}_i$, kNN simply uses the distance from the instance to its k^{th} nearest neighbor:

$$\text{kNN}_k(\mathbf{x}_i) = k\text{-dist}(\mathbf{x}_i) \quad (2.13)$$

kNN-weight

kNN-weight (kNNW) (ANGIULLI; PIZZUTI, 2002; ANGIULLI; PIZZUTI, 2005) is a kNN variant which uses the sum of distances from the instance to its *k* nearest neighbors, instead of using only the distance to its k^{th} nearest neighbor. The use of the sum of the distances makes the score less sensitive to a change of the parameter *k*. The sum of the distances can be normalized by *k* to make the values more comparable across different parameterizations:

$$\text{kNNW}_k(\mathbf{x}_i) = \frac{1}{k} \sum_{j=1}^k j\text{-dist}(\mathbf{x}_i) \quad (2.14)$$

As most unsupervised outlier detection algorithms, k NN and k NNW return only a score for each instance. The higher the score, the greater the evidence of being an outlier. If one wants a binary classification between inliers and outliers, the top- n instances with the highest scores are labeled as outliers and the remaining as inliers.

LDOF

Local Distance-based Outlier Factor (LDOF) (ZHANG; HUTTER; JIN, 2009) is a distance-based algorithm that normalizes the k NNW values by the average pairwise distances within the k NN set:

$$\text{LDOF}_k(\mathbf{x}_i) = \frac{\text{kNNW}_k(\mathbf{x}_i)}{\frac{1}{k(k-1)} \sum_{\mathbf{x}_p, \mathbf{x}_q \in k\text{NN}(\mathbf{x}_i)} d(\mathbf{x}_p, \mathbf{x}_q)} \quad (2.15)$$

The normalization of the scorings by the pairwise distances makes the algorithm local rather than global. A LDOF value of approximately 0.5 indicates that the instance is located within a uniform region, *i.e.*, it is not an outlier. The higher the LDOF value, the more it is considered an outlier.

ODIN

Outlier Detection using Indegree Number (ODIN) (HAUTAMÄKI; KÄRKKÄINEN; FRÄNTI, 2004) utilizes a k NN graph to measure the characterization of each instance as outlier. A k NN graph is a weighted directed graph, in which every vertex represents an instance of the dataset, and a directed edge from $\mathbf{x}_i$ to $\mathbf{x}_j$ represents that $\mathbf{x}_j$ is a nearest neighbor of $\mathbf{x}_i$. The weights of the edges represent the distance between the instances. Unlike the other distance methods discussed, ODIN considers outliers instances that are far away from their Reverse k NNs (Rk NNs) (RADOVANOVIC; NANOPOULOS; IVANOVIĆ, 2015), such that the outlier score of a given instance $\mathbf{x}_i$ is the number of incident edges in that vertex (its indegree) in the k NN graph:

$$\text{ODIN}_k(\mathbf{x}_i) = |Rk\text{NN}(\mathbf{x}_i)| \quad (2.16)$$

In contrast to the other unsupervised outlier detection algorithms, outliers are expected to have low scores, since the algorithm considers outliers instances with few Rk NN.

2.3.3 Density Methods

In order to judge quantitatively whether an instance should be considered outlier or not, density-based techniques estimate the density of each instance $\mathbf{x}_i$. These techniques assume that instances in low density regions are outliers, while instances in dense regions are inliers. Note that unlike semi-supervised density methods, unsupervised density methods calculate the density

of each instance $\mathbf{x}_i$ instead of density of the inlier class. Most density methods use the distances from the instance to their nearest neighbors as a basis for estimating the inverse of its density. Among the most well-known density methods are LOF, SimplifiedLOF, COF, INFLO, LDF, KDEOS, LOCI, and GLOSH, presented in the following.

LOF

Local Outlier Factor (LOF) (BREUNIG *et al.*, 2000) is the first density-based method among the local outlier detection methods. Since its introduction, many other variants have been proposed following somehow the same intuition (SCHUBERT; ZIMEK; KRIEGEL, 2014b). LOF compares the local density of an instance to that of its neighbors. In order to avoid effects similarly to masking and swapping, the distances between instances are replaced by reachability distances, defined as:

$$\text{reach-dist}_k(\mathbf{x}_i \leftarrow \mathbf{x}_j) = \max\{k\text{-dist}(\mathbf{x}_j), d(\mathbf{x}_i, \mathbf{x}_j)\} \quad (2.17)$$

The local reachability density (lrd) of an instance $\mathbf{x}_i$ is then defined as the inverse average reachability distance from the set of $\mathbf{x}_i$'s neighbors, $k\text{NN}(\mathbf{x}_i)$, that are within the k nearest neighbor distance around $\mathbf{x}_i$:

$$\text{lrd}_k(\mathbf{x}_i) = \frac{|k\text{NN}(\mathbf{x}_i)|}{\sum_{\mathbf{x}_j \in k\text{NN}(\mathbf{x}_i)} \text{reach-dist}_k(\mathbf{x}_i \leftarrow \mathbf{x}_j)} \quad (2.18)$$

Finally, the LOF score of an instance is computed by comparing the lrd of the instance with that of its neighbors:

$$\text{LOF}_k(\mathbf{x}_i) = \frac{\sum_{\mathbf{x}_j \in k\text{NN}(\mathbf{x}_i)} \frac{\text{lrd}_k(\mathbf{x}_j)}{\text{lrd}_k(\mathbf{x}_i)}}{|k\text{NN}(\mathbf{x}_i)|} \quad (2.19)$$

A LOF value of approximately 1 indicates that the instance has, on average, approximately the same density as its k nearest neighbors. A LOF value less than 1, the instance has, on average, a density higher than its k nearest neighbors. In both cases, the instance can be considered inlier. Contrarily, a high LOF value suggests that the instance is in a low density region compared to its k nearest neighbors. Therefore, the higher the score, the higher the evidence that the instance is an outlier.

SimplifiedLOF

SimplifiedLOF (SCHUBERT; ZIMEK; KRIEGEL, 2014b; CAMPOS *et al.*, 2016) is an (unintentional) adaptation of LOF by (mistakenly) replacing $k\text{-dist}(\mathbf{x}_j)$ with $k\text{-dist}(\mathbf{x}_i)$ in Equation (2.17). Since $\mathbf{x}_j \in k\text{NN}(\mathbf{x}_i)$ and $d(\mathbf{x}_i, \mathbf{x}_j) \leq k\text{-dist}(\mathbf{x}_i)$ by definition, it makes the

$\text{reach-dist}_k(\mathbf{x}_i \leftarrow \mathbf{x}_j)$ to be replaced by only $k\text{-dist}(\mathbf{x}_i)$, which results in a simpler density estimate:

$$\text{dens}_k(\mathbf{x}_i) = \frac{1}{k\text{-dist}(\mathbf{x}_i)} \quad (2.20)$$

Despite the simplification of the density estimate, the computational complexity of SimplifiedLOF is similar to LOF.

COF

Connectivity-based Outlier Factor (COF) (TANG *et al.*, 2002) is one of the many LOF variants that modify the density estimation. COF modifies the density estimation to account for the concept of isolativity, which refers to the degree that an instance is “connected” to other instances. The “connectedness” of an instance $\mathbf{x}_i$ is measured via a Minimum Spanning Tree (MST) of its neighborhood rooted at $\mathbf{x}_i$. The MST is reduced back to the root $\mathbf{x}_i$ by iteratively deleting the leaf node incident to the edge having greatest length. The average chaining distance of $\mathbf{x}_i$ is then achieved by computing the total edge lengths for each of the MSTs in the iterations, and then averaging over the MSTs:

$$\text{ac-dist}_k(\mathbf{x}_i) = \sum_{j=1}^k \frac{2(k+1-j)}{k(k+1)} \delta(e_j) \quad (2.21)$$

where e_{k-j} is the edge deleted in the j^{th} iteration, and $\delta(e_{k-j})$ is its length. The COF value of $\mathbf{x}_i$ is then given as a ratio of the ac-dist of $\mathbf{x}_i$ with the average of the ac-dist values of its k nearest neighbors:

$$\text{COF}_k(\mathbf{x}_i) = \frac{\text{ac-dist}_k(\mathbf{x}_i)}{\frac{1}{k} \sum_{\mathbf{x}_j \in k\text{NN}(\mathbf{x}_i)} \text{ac-dist}_k(\mathbf{x}_j)} \quad (2.22)$$

INFLO

Influenced Outlierness (INFLO) (JIN *et al.*, 2006) is a LOF variant which diverges from most local algorithms by defining the context set differently from the reference set. While the context set is the set of instances used to compute the quantities (*e.g.*, densities or distances to nearest neighbors) related to the instances, the reference set is the set used to compare these quantities. INFLO uses the same density estimate as SimplifiedLOF (Equation (2.20)). Therefore, the context set is composed only by the $k\text{NN}$. The reference set, however, contains the union of $k\text{NN}$ and $Rk\text{NN}$:

$$\text{INFLO}_k(\mathbf{x}_i) = \frac{\sum_{\mathbf{x}_j \in (k\text{NN}(\mathbf{x}_i) \cup Rk\text{NN}(\mathbf{x}_i))} \frac{\text{dens}_k(\mathbf{x}_j)}{\text{dens}_k(\mathbf{x}_i)}}{|k\text{NN}(\mathbf{x}_i) \cup Rk\text{NN}(\mathbf{x}_i)|} \quad (2.23)$$

LoOP

Local Outlier Probabilities (LoOP) (KRIEGEL *et al.*, 2009) is a LOF variant which normalizes the outlier detection scores in the range of $[0, 1]$ that is directly interpretable as the probability that a data instance is an outlier. LoOP modifies the density estimate by assuming that the local distances from $\mathbf{x}_i$ to its k NN approximate a half-Gaussian distribution centered in $\mathbf{x}_i$. Hence, LoOP uses a density estimate based on the quadratic mean distance:

$$LoOP-dens(\mathbf{x}_i) = 1/\sqrt{\frac{\sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} d(\mathbf{x}_i, \mathbf{x}_j)^2}{|kNN(\mathbf{x}_i)|}} \quad (2.24)$$

Using the Gaussian Error Function, LoOP normalizes the outlier detection scores, such that the LoOP value will be close to 0 for instances within dense regions and close to 1 for density based outliers.

LDF

Local Density Factor (LDF) (LATECKI; LAZAREVIC; POKRAJAC, 2007) is a LOF variant which replaces LOF's density estimate by a Gaussian Kernel Density Estimation (KDE) modified to use the reachability distance (Equation (2.17)):

$$LDE_{(k,h)}(\mathbf{x}_i) = \frac{1}{|kNN(\mathbf{x}_i)|} \sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} \frac{1}{(2\pi)^{d/2} (h \cdot k-dist(\mathbf{x}_i))^d} e^{-\frac{1}{2} \frac{reach-dist_k(\mathbf{x}_i \leftarrow \mathbf{x}_j)}{(h \cdot k-dist(\mathbf{x}_i))^2}} \quad (2.25)$$

The LDE is then compared to the LDE of its neighbors:

$$LDF_{(k,h,c)}(\mathbf{x}_i) = \frac{\frac{1}{k} \sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} LDE_{(k,h)}(\mathbf{x}_j)}{LDE_{(k,h)}(\mathbf{x}_i) + \frac{c}{k} \sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} LDE(\mathbf{x}_j)} \quad (2.26)$$

where c is a scaling constant to normalize the LDF values on the scale from 0 to $1/c$.

KDEOS

Kernel Density Estimation Outlier Score (KDEOS) (SCHUBERT; ZIMEK; KRIEGEL, 2014a) is a LOF variant which also uses KDE into the density estimation, but unlike LDE, it keeps the mathematical kernel density estimation intact. The KDE density is compared with those in the k NN set by the z -score over a range of neighborhood sizes from $k_{\min}$ to $k_{\max}$:

$$s(\mathbf{x}_i)_{(k_{\min}, k_{\max})} = \text{mean}_{k_{\min} \dots k_{\max}} z\text{-score}(KDE_k(\mathbf{x}_i), \{KDE_k(\mathbf{x}_j)\}_{\mathbf{x}_j \in kNN(\mathbf{x}_i)}) \quad (2.27)$$

Furthermore, the authors assume that the resulting scores $s(\mathbf{x}_i)$ are approximately normally distributed to use the normal cumulative density function to normalize the scores to the range $[0, 1]$.

LOCI

Local Correlation Integral (LOCI) (PAPADIMITRIOU *et al.*, 2003) is a LOF variant which can find both local and global outliers by determining the reference set for each instance dynamically rather than using a fixed neighborhood size. LOCI determines a set of relevant radii $\mathcal{R}$ and analyzes the density of an instance at multiple neighborhood radii αr of a given maximum radius $r \in \mathcal{R}$, where $\alpha \in (0, 1]$. For each instance $\mathbf{x}_i$, a reference set $\mathcal{N}(\mathbf{x}_i, r) = \{\mathbf{x}_j | d(\mathbf{x}_i, \mathbf{x}_j) \leq r\}$ and a (local) density $n(\mathbf{x}_i, r) = |\mathcal{N}(\mathbf{x}_i, r)|$ are defined.

The average density inside the reference set of an instance $\mathbf{x}_i$ is then defined as:

$$\hat{n}_\alpha(\mathbf{x}_i, r) = \frac{\sum_{\mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i, r)} n(\mathbf{x}_j, \alpha r)}{n(\mathbf{x}_i, r)} \quad (2.28)$$

The density of the instance is compared to average density inside the reference set using the Multi-Granularity Deviation Factor (MDEF):

$$\text{MDEF}_\alpha(\mathbf{x}_i, r) = 1 - \frac{n(\mathbf{x}_i, \alpha r)}{\hat{n}_\alpha(\mathbf{x}_i, r)} \quad (2.29)$$

The LOCI score is then computed as the MDEF normalized by the standard deviation with respect to the MDEF of the instances in the reference set:

$$\text{LOCI}_\alpha(\mathbf{x}_i) = \max_{r \in \mathcal{R}} \left\{ \frac{\text{MDEF}_\alpha(\mathbf{x}_i, r)}{\sigma_{\text{MDEF}_\alpha}(\mathbf{x}_i, r)} \right\} \quad (2.30)$$

The authors suggest to label as outliers instances with LOCI values greater than 3, as they deviate 3 times from the normalized deviation of the MDEF values of their reference set.

GLOSH

The Global-Local Outlier Scores from Hierarchies (GLOSH) (CAMPELLO *et al.*, 2015) is an unsupervised outlier detection algorithm based on the hierarchical density estimates provided by the hierarchical clustering algorithm HDBSCAN*. Similarly to LOCI, GLOSH can also find global and local outliers, but instead of using a spherical neighborhood (which can lead to comparing outliers to other outliers or unrelated clusters), GLOSH determines the neighborhood for each instance based on the closest cluster (from a density-based perspective) within the density-based hierarchy. After a density-based clustering hierarchy is computed for the whole dataset, the GLOSH score for each instance $\mathbf{x}_i$ can be computed based on the difference in density around $\mathbf{x}_i$ and the highest density inside the cluster closest to $\mathbf{x}_i$ in the HDBSCAN* hierarchy, as follows:

$$\text{GLOSH}(\mathbf{x}_i) = \frac{\lambda_{\max}(C_{\mathbf{x}_i}) - \lambda(\mathbf{x}_i)}{\lambda_{\max}(C_{\mathbf{x}_i})} \quad (2.31)$$

where $\lambda(\mathbf{x}_i)$ is the density of $\mathbf{x}_i$ and $\lambda_{\max}(C_{\mathbf{x}_i})$ is the highest density of an instance inside the closest cluster $C_{\mathbf{x}_i}$, where densities are estimated by a k -nearest neighbor density estimator. The closest cluster $C_{\mathbf{x}_i}$ is the one that $\mathbf{x}_i$ belongs to at the density level of $\mathbf{x}_i$.

The GLOSH score ranges in the interval $[0, 1)$, where a score of 0 means the density of an instance $\mathbf{x}_i$ is equivalent to that of the densest instance in $C_{\mathbf{x}_i}$, while a score close to 1 means that $\mathbf{x}_i$ has a much lower density than $C_{\mathbf{x}_i}$, therefore, it is probably an outlier.

2.3.4 Angle Methods

Angle-based methods represent each instance of the dataset as a point in Euclidean space, which restricts these methods only to numeric datasets. The intuition behind these methods is that by measuring the variance in the angles between an instance and pairs of other instances, we can determine whether or not an instance is an outlier. If the variance is high, it suggests that the instance is surrounded by other instances (in a cluster), while a low variance suggests that the instance is far away from other instances (an outlier). The advantage of this approach is the fact of using the angle of the points, rather than a measure based on distance, makes these methods more robust when applied to the high-dimensional datasets.

ABOD

The Angle-Based Outlier Detection (ABOD) (KRIEGEL; SCHUBERT; ZIMEK, 2008) is an global outlier detection algorithm which proposes not only to use the distance between points but primarily the variance of the angles between points. ABOD computes the variance of the angles between point $\mathbf{x}_i$ and all other pairs of points in the dataset $\mathbf{X}$, weighted by the inverse of the distances to the respective points. This weighting factor is important since the angle to a pair of points varies naturally more for a larger distance. Formally, the Angle-Based Outlier Factor (ABOF) is defined as follows:

$$\text{ABOF}(\mathbf{x}_i) = \text{VAR}_{\mathbf{x}_p, \mathbf{x}_q \in \mathbf{X}} \left(\frac{\langle \mathbf{x}_i - \mathbf{x}_p, \mathbf{x}_i - \mathbf{x}_q \rangle}{\|\mathbf{x}_i - \mathbf{x}_p\|^2 \cdot \|\mathbf{x}_i - \mathbf{x}_q\|^2} \right) \quad (2.32)$$

In contrast to most unsupervised outlier detection algorithms, outliers are expected to have low scores, since low angle variance is expected for points considered outliers.

FastABOD

FastABOD (KRIEGEL; SCHUBERT; ZIMEK, 2008) was designed as a fast variant of ABOD. Since the complete computation of the angles to all other pairs of points requires $\mathcal{O}(N^3)$ time, FastABOD considers only those pairs of points that are among the k NNs, which requires only $\mathcal{O}(k^2)$ computations per point.

2.4 Chapter Remarks

In this chapter, we presented the basic concepts of outlier detection algorithms. We categorize such algorithms into supervised, semi-supervised, and unsupervised techniques. We gave special emphasis to semi-supervised and unsupervised algorithms, since they are the main focus of this thesis.

The supervised outlier detection may be seen as a special case of imbalanced binary classification. We briefly presented two approaches to tackle this problem: Cost Sensitive Learning (at the algorithmic level) and Sampling (at the data level).

In the semi-supervised outlier detection, we are only provided with previously known inliers and the (one-class) classification model has to discriminate the inlier class. We categorized such models into density methods, boundary methods, and reconstruction methods. For the density methods, we described two algorithms: Gaussian Density (parametric) and PW (non-parametric). For the boundary methods, we described four algorithms: OC-SVM, SVDD, LP, and kNNDD. OC-SVM and SVDD are both inspired in SVM. We discussed the relation between them and showed that OC-SVM and SVDD find the same decision boundary if the conditions are met. For the reconstruction methods, we described the Auto-Encoder Network.

In unsupervised outlier detection, we are provided with a set of unlabeled instances and the algorithm has to determine whether each instance is an inlier or an outlier. We discussed two categorizations for the unsupervised detection algorithms. A categorization regarding the scope considered by the algorithms to decide whether the instance is an outlier (local vs. global). We presented different approaches to defining such scope: global (*e.g.*, kNN and ABOD), fixed-local (*e.g.*, LOF and many variants), dynamic-local based on a radius (LOCI) and dynamic-local based on clusters (GLOSH). Another categorization discussed is regarding the approaches used by the algorithms, which goes from the classic parametric statistical algorithms to more recent database-oriented approaches (distance, density, and angle).

A comparative study between one-class classification algorithms was conducted by [Tax \(2001\)](#). For unsupervised outlier detection, [Campos *et al.* \(2016\)](#) performed such a comparative study. In the next chapter, we present a framework to adapt unsupervised outlier detection methods to the one-class classification problem. Some of the unsupervised algorithms that we discussed in this chapter are adapted, and a comparative study between one-class classification algorithms and such adapted unsupervised outlier detection methods is performed.

ON THE EVALUATION OF OUTLIER DETECTION AND ONE-CLASS CLASSIFICATION METHODS

It has been shown that unsupervised outlier detection methods can be adapted to the one-class classification problem. Following the approach proposed in (JANSSENS; FLESCHE; POSTMA, 2009), the unsupervised outlier detection methods can be extended to use inlier class information to be applicable also in the semi-supervised setting.

In this chapter, we focus on the comparison of one-class classification algorithms with such adapted unsupervised outlier detection methods, improving on previous comparison studies in several important aspects. We study a number of one-class classification and unsupervised outlier detection methods in a rigorous experimental setup, comparing them on a large number of datasets with different characteristics, using different performance measures. Our experiments led to conclusions that do not fully agree with those of previous work.

The comparative study presented in this chapter is an extension of the article “*On the Evaluation of Outlier Detection and One-Class Classification Methods*” (DSAA’2016) (SWERSKY *et al.*, 2016). The adaptation of GLOSH to the semi-supervised setting, as described in this chapter, was originally conceived by Swersky (2018). A more in-depth discussion on the GLOSH adaptation can be found in (SWERSKY, 2018), which also describes the original study published in DSAA’2016 (SWERSKY *et al.*, 2016). The author of this thesis participated in this study (and as co-author in the aforementioned 2016 publication) by programming and adapting part of the codes, running the experiments, discussing and analyzing the results. In this chapter, we extended this work by including synthetic datasets, measuring the performance also with respect to MCC, as well as discussing and analyzing the conclusions arising from four new experimental scenarios.

3.1 Introduction

Although both unsupervised outlier detection and one-class classification were first studied in the statistics field ([BARNETT; LEWIS, 1994](#); [MARKOU; SINGH, 2003](#)), little has been done in the literature in order to compare the performance of both categories of algorithms. The one-class classification scenario generally is an easier task, because training data for one class is available. While in the one-class classification setting one can estimate the probability density function or other models without considering the presence of outliers in the dataset, in the unsupervised outlier detection setting one must deal with possible outliers while estimating the probability density function or other models. Given the differences between these two settings, the corresponding methods cannot be compared in a straightforward way.

[Janssens and Postma \(2009\)](#) proposed a methodological framework to make unsupervised outlier detection algorithms work in a one-class classification setup and thus be able to compare them against algorithms specifically designed for this task. The authors, however, only assessed the performance of two unsupervised methods, namely, LOF and LOCI, in the one-class classification scenario. In a follow-up work ([JANSSENS; FLESCHE; POSTMA, 2009](#)), the same framework previously proposed in ([JANSSENS; POSTMA, 2009](#)) was once again applied to LOF and LOCI, but this time these methods were also compared against three one-class classification methods (PW, SVDD, and kNNDD). The authors concluded that LOF and SVDD are the top two performers with an identical average performance rank, although each method has particular scenarios where one may outperform the other.

In this chapter, we perform a more comprehensive investigation, using a more rigorous experimental setup, which leads to conclusions that do not fully agree with [Janssens, Flesch and Postma \(2009\)](#). In particular, we make the following contributions:

- When reproducing the experiments in ([JANSSENS; FLESCHE; POSTMA, 2009](#)), which reports averages over 5 repetitions of 5-fold cross-validation, we noticed a large variability in the results. In order to increase results confidence, we perform 30 repetitions using 10-fold cross-validation instead.
- We increase the number of datasets used in the evaluation from 24 to 101 (31 real datasets plus 70 synthetic datasets), and the number of compared methods from 5 to 11.
- In addition to the well-known Area Under the ROC curve (ROC AUC), used in [Janssens, Flesch and Postma \(2009\)](#), we also use Matthews Correlation Coefficient (MCC) to measure performance. These measures complement each other and together give a more complete picture of the performance characteristic of a method.
- In addition to the only experimental scenario performed by [Janssens, Flesch and Postma \(2009\)](#), we also evaluated the algorithms in five additional experimental scenarios. All of

them represent possible real application scenarios. We show that some of the conclusions change depending on the application scenarios.

- We also include and describe an adaptation of a recent outlier detection method — called GLOSH (CAMPELLO *et al.*, 2015) — to the one-class classification problem.
- Last but not least, we discuss basic principles of one-class classification that should not be violated when adapting unsupervised outlier detection methods to this task. We carefully examined and adjusted all the codes used in our experiments, making sure that these principles were not violated when producing the reported results.

The remainder of this chapter is organized as follows. In section 3.2 we present the methodological framework to adapt unsupervised outlier detection methods to the one-class classification problem and discuss basic principles of one-class classification that should not be violated when performing such adaptation. We describe the setup of the experiments in section 3.3 and discuss the results in section 3.4. Finally, we summarize and conclude the chapter in section 3.5.

3.2 Adapting Unsupervised Outlier Detection Methods to the One-Class Classification Setting

Common to unsupervised outlier detection methods is that they compute a certain outlier score for each instance. To use an unsupervised outlier detection method in one-class classification, the general strategy is as follows: First run the unsupervised method on the (one-class) training data, pre-computing the scores for each inlier (Figure 2a). Then compute the score for a new instance to be classified (Figure 2b), possibly using other pre-computed quantities (*e.g.*, densities or distances to nearest neighbors) related to instances in the training data. Then, in order to classify the new instance, compare its score with the pre-computed scores for the inliers. In the case of the example in Figure 2, the new instance is classified as outlier, as its score is higher than any pre-computed scores for the inliers.

When classifying multiple instances, it is also recommended that the classification procedure described above be performed independently for each instance. This way, different unlabeled instances will not affect each other's assessment. The reason we only classify one instance at a time instead of multiple instances at once is because we make no assumptions about the nature of each instance in relation to the combined dataset as a whole. It is possible that instances to be classified, while outliers in the sense that they do not belong to the inlier class, may be grouped together in such a way that an unsupervised method would not detect them as outliers.

Figure 2 – Illustrative example of the framework for adapt unsupervised outlier detection methods to one-class classification. The training instances in black crosses. The new instance to be classified in red cross. The circle around the crosses represent their respective outlier scores. The large the circle, the higher the outlier score.

There are two important aspects related to the use of pre-computed quantities (*e.g.*, densities or distances to nearest neighbors) that involve training data when computing the outlier score for a new instance to be classified. First, when classifying multiple instances, there is no need to recompute these quantities over and over again for each new instance, since they relate solely to the training data (inlier class model) and can thus be pre-computed, which makes computations faster.

Second and more importantly, using pre-computed quantities regarding the instances in the training data assures that the model is by no means affected by new instances to be classified. This is a basic principle of one-class classification, *i.e.*, unlabeled instances should not affect the pre-computed inlier class model, since they may be outliers. For example, suppose that a certain algorithm operates by comparing the density of a new instance to be classified against the densities of its nearest neighbors among the known inliers (training data). In this case, the densities of the inliers should be pre-computed, not to be affected by the presence of the unlabeled instance being currently assessed; otherwise, each unlabeled instance would affect the model in a different way, which means that different instances would be classified by different models/criteria.

In general, the violation of the second aspect occurs in local outlier detection methods, as these methods compare the instance to be classified with a reference set. Therefore, one needs to be careful to adapt these methods, as the quantities of the reference set can be affected by the instance to be classified, in case of re-computation. In the following, we discuss in more detail the adaptation of LOF and its variants, and GLOSH to the semi-supervised setting.

3.2.1 Adapting LOF and its Variants to One-Class Classification

The operation of LOF and its variants to compute the outlier score of an instance can be summarized in 4 steps:

1. Define the reference set. For the variants discussed in chapter 2, the reference set is typically composed by the k NN. However, we also discussed a variant that includes the Rk NN (INFLO) and another that defines the reference set as the instances within a radius r (LOCI).
2. Define the context set for each instance in the reference set. LOF and its variants usually use the same approach to define the instances of the reference set and the context set, but there are variants which define the sets differently. For example, INFLO uses only the k NN in the context set but the union of k NN and Rk NN in the reference set. LOCI, which defines the reference set as the instances within a radius r , uses only the instances within a fraction α of the radius r in the context set.
3. Estimate the density of each instance in the reference set based on its context set. Each of the LOF variants uses its own notion of density, from the most simple, for example, using the inverse of the k -dist($\cdot$) (SimplifiedLOF) or just counting the number of instances in the context set (LOCI), to the most complex, for example, using a Kernel Density Estimation (LDF and KDEOS) or computing the connectivity via MST (COF).
4. Finally, compare the density of the instance with those in the reference set. The usual way to compare the densities is by the ratio between the instance density and the reference set density, such that values close to 1 indicate that the instance is in a region of the same density than the reference set. Other approaches involve normalization, for example, using the z -score (KDEOS), the standard deviation (LOCI), or transforming the values into probabilities (LoOP).

When framing LOF and its variants into the semi-supervised setting, the training phase consists of pre-computing the scores for the training set following the 4 steps discussed above. These scores will be used to compare against the scores of new instances to be classified, such that outlier scores higher than those in the training set are classified as outliers. In addition, the density estimation computed in step 3 should be stored to be reused in the test phase. A re-computation in the test phase, besides being inefficient, may cause this quantity to be affected by the instance to be classified. It may occur in step 2 when the new instance can be mistakenly included in the context set of some instance of the reference set, distorting the density estimate of this instance.

In Figure 3, we show an example where the new instance to be classified ($\mathbf{x}_i$) affects the density estimation of an instance belonging to its reference set ($\mathbf{x}_j$). In Figure 3a, the density

estimation of $\mathbf{x}_j$ is computed using only the training data and the new instance is classified as outlier. In Figure 3b, we consider the new instance part of the dataset when computing the density estimation of the reference set, and it becomes one of the nearest neighbors of $\mathbf{x}_j$. Then, density estimation of $\mathbf{x}_j$ is affected by the new instance, decreasing its original value and causing a similar effect to masking, classifying the new instance as inlier.

Figure 3 – Illustrative example of how the new instance may affect the density estimation computation of the reference set. The training instances in black dots. The new instance to be classified in red diamond. Instances within the dashed circle are the 5 nearest neighbors of the instance in the center.

3.2.2 Adapting GLOSH to One-Class Classification

Unlike the other local outlier detection methods, GLOSH defines the reference sets of each instance dynamically based on the closest structure within the HDBSCAN* density hierarchy. To apply GLOSH in a one-class classification scenario, we can construct initially the HDBSCAN* hierarchy using the training data, and then use this hierarchy as a fixed “model” to compute outlier scores for unseen data. In order to classify a new instance $\mathbf{x}_i$, we must determine which is the closest cluster $C_{\mathbf{x}_i}$ in the fixed hierarchy. This can be achieved by first adding a given instance $\mathbf{x}_i$ to the Minimum Spanning Tree (MST) which underlies the HDBSCAN* hierarchy; $\mathbf{x}_i$ is connected to the training instance $\mathbf{x}_j$ with the smallest “distance” in the density space in which the MST is constructed. We can find the closest cluster $C_{\mathbf{x}_i}$ for an instance $\mathbf{x}_i$ by removing the edges of the MST in decreasing order of weight. The closest cluster $C_{\mathbf{x}_i}$ is the one that $\mathbf{x}_i$ belongs before being detached from the MST.

There are two situations that may occur when connecting the new instance $\mathbf{x}_i$ to the

training instance $\mathbf{x}_j$. The first situation is that the edge connecting $\mathbf{x}_i$ and $\mathbf{x}_j$ is removed before $\mathbf{x}_j$ is separated from the MST. In this case, the closest cluster to $\mathbf{x}_i$ will be the cluster that $\mathbf{x}_j$ belongs to when $\mathbf{x}_i$ is detached. We show an example of this situation in Figure 4.

Figure 4 – MST which underlies the HDBSCAN* hierarchy in the density space with minimum cluster size (m_{clSize}) = 3. The new instance $\mathbf{x}_i$ is connected to the training instance with the smallest distance $\mathbf{x}_5$ (Figure 4a). The edge connecting $\mathbf{x}_i$ and $\mathbf{x}_5$ is removed before $\mathbf{x}_5$ is separated from the MST and so the closest cluster to $\mathbf{x}_i$ is the one $\mathbf{x}_5$ is assigned at this density level. In this case, it is the cluster where all instances remain clustered (Figure 4b). Figure based on (SWERSKY, 2018).

The second situation is that $\mathbf{x}_j$ is separated from the MST before the edge between $\mathbf{x}_i$ and $\mathbf{x}_j$ is removed. In this case, the closest cluster for $\mathbf{x}_i$ is the same as the closest cluster for $\mathbf{x}_j$. For this situation, we show an example in Figure 5.

Figure 5 – MST which underlies the HDBSCAN* hierarchy in the density space with minimum cluster size (m_{clSize}) = 3. The new instance $\mathbf{x}_i$ is connected to the training instance with the smallest distance $\mathbf{x}_5$ (Figure 5a). The edge connecting $\mathbf{x}_5$ is separated from the MST before the edge between $\mathbf{x}_i$ and $\mathbf{x}_5$ (Figure 5c) and so the closest cluster to $\mathbf{x}_i$ is the closest cluster for $\mathbf{x}_5$. In this case, it is the cluster with the instances $\mathbf{x}_4, \mathbf{x}_5$ and $\mathbf{x}_6$ (Figure 5b). Figure based on (SWERSKY, 2018).

Once we have identified the closest cluster for $\mathbf{x}_i$, we can compute its GLOSH score using Equation (2.31).

3.3 Experimental Setup

We compare and evaluate the 6 one-class classification algorithms described in the previous chapter: Gaussian Density, Parzen Window (PW), Support Vector Data Description (SVDD), Linear Programming (LP), k -Nearest Neighbor Data Description (which we call here $\text{kNN}_{\text{local}}$), and Auto-Encoder, against 5 unsupervised outlier detection algorithms adapted to one-class classification: k -Nearest Neighbor (which we call here $\text{kNN}_{\text{global}}$), Local Outlier Factor (LOF), Local Correlation Integral (LOCI), Global-Local Outlier Scores from Hierarchies (GLOSH), and Angle-Based Outlier Detection (ABOD). We use code from the repository available at <http://prlab.tudelft.nl/users/david-tax/> (TAX, 2018) for most compared algorithms, except for LOF, LOCI, $\text{kNN}_{\text{local}}$, and GLOSH. In the case of LOF and LOCI, their implementations were modified to ensure that new instances to be classified do not affect the pre-computed model for the inlier class, following the guidelines previously discussed in section 3.2. As $\text{kNN}_{\text{local}}$ was not available in that repository, we used our own implementation for this algorithm. GLOSH was adapted based on the implementation of HDBSCAN* made available at <http://lapad-web.icmc.usp.br/>.

3.3.1 Datasets

We use three different collections of previously published benchmarking of datasets to evaluate the results of our experiments. A collection of low dimensional synthetic datasets (HANDL; KNOWLES; KELL, 2005), a collection of high dimensional synthetic datasets (ZIMEK *et al.*, 2013; ZIMEK; CAMPELLO; SANDER, 2014), and a collection of real-world datasets containing both high and low dimensional datasets (DHEERU; TANISKIDOU, 2017).

For experiments on synthetic data, we adopt previously published benchmarking datasets. The first collection of synthetic data consists of 40 2-dimensional datasets provided by Handl, Knowles and Kell (2005), comprising examples with 4, 10, 20, and 40 clusters. The size of the clusters is uniformly chosen from $[50, 500]$ for datasets with 4 and 10 clusters and from $[10, 100]$ for datasets with 20 and 40 clusters. This dataset collection has been designed to evaluate clustering results, but it has also been used to evaluate outliers (MARQUES *et al.*, 2015).

In addition to the previous collection of synthetic datasets, we also use a second collection of synthetic datasets to evaluate the methods' performance on high dimensionality. This second collection (30 datasets) has been designed and used to evaluate outlier detection methods. We use a collection referred to as “*batch1*” (ZIMEK *et al.*, 2013; ZIMEK; CAMPELLO; SANDER, 2014). The 30 datasets in this collection vary in the dimensionality $d \in [20, \dots, 40]$, in the number of clusters $c \in [2, \dots, 10]$, and for each cluster independently in the number of points $n_{c_i} \in [600, \dots, 1000]$. For each cluster, the points are generated following a Gaussian model with randomly selected parameters that are attribute-wise independent. The sampled cluster is

randomly rotated and the covariance matrix is rotated accordingly.

In addition to the collections of synthetic datasets, we use 31 real-world datasets from the UCI Machine Learning Repository (DHEERU; TANISKIDOU, 2017) as pre-processed for one-class classification and made available at <http://prlab.tudelft.nl/users/david-tax/>: Abalone, Arrhythmia, Balance-scale, Ball-bearing, Biomed, Breast, Cancer, Colon, Delft1x3, Delft2x2, Delft3x2, Delft5x1, Delft5x3, Diabetes, Ecoli, Glass, Heart, Hepatitis, Housing, Imports, Ionosphere, Iris, Liver, Satellite, Sonar, Spectf, Survival, Vehicle, Vowels, Waveform and Wine.

In total, we have 101 multi-class datasets. We average the results for the Delft datasets, for they are variants obtained from the same source. Finally, due to the inability of some algorithms to deal with replicated instances, duplicates are removed from the datasets where they are present.

3.3.2 Labeling of Datasets

In order to consider the different one-class classification scenarios, we perform three major types of experiments w.r.t. the labeling of datasets. In the **first type** of experiment (Type I), we follow the only approach taken by Janssens, Flesch and Postma (2009), where multi-class datasets are transformed into one-class datasets by re-labeling one class as inliers, and the remaining classes as outliers. Except for datasets where only a single inlier class has been pre-defined as such in the data repository (<http://prlab.tudelft.nl/users/david-tax/>) — *e.g.*, Ecoli — we repeat the procedure for every class as inliers in a dataset, and average the results.

For the **second type** of experiment (Type II), we reverse the inlier and outlier classes obtained in the first type of experiment for datasets that have more than 2 possible inlier classes defined. This type of experiment is important since it models situations with a possibly multi-modal inlier class. Note that Type II experiments are only different from Type I — and are therefore only reported — for datasets with 3 or more classes.

In the **third type** of experiment (Type III), we use the Mahalanobis distance to label the instances in the border of the classes as local outliers. Based on the covariance matrix, the Mahalanobis distance between the mean of a class and each class instance is computed. The distribution of the Mahalanobis distance follows a χ_d^2 distribution with d degrees of freedom. Those points that exhibit a distance to their class center larger than the theoretical 0.975 quantile were labeled as outliers, independently of the actually occurring Mahalanobis distances of the sampled points. This results in an expected amount of 2.5% outliers per dataset. For the collection of high-dimensional synthetic datasets (ZIMEK *et al.*, 2013; ZIMEK; CAMPELLO; SANDER, 2014), the authors provided this labeling based on the original class covariance matrices. However, for the first collection (HANDL; KNOWLES; KELL, 2005), the original class covariance matrices were unknown and therefore we calculated from the data to label the instances.

In Figure 6 is shown one labeling example for each of the three major types of experiments with respect to the labeling of the datasets.

Figure 6 – Three major types of experiments w.r.t. the labeling of datasets. Inliers marked in black dots and outliers marked in red diamonds.

In addition to the experiments regarding the labeling of the dataset, we also evaluate the behavior of the methods in three other scenarios. In the **fourth type** of experiment (Type IV), we use the synthetic datasets with less than 20 instances to evaluate the performance on datasets with small sample size. Similarly, the synthetic datasets with more than 20 instances are used to evaluate the performance on datasets with large sample size in the **fifth type** of experiment (Type V). Finally, in the **sixth type** of experiment (Type VI), we use the collection of high-dimensional synthetic datasets to evaluate the methods' performance on high dimensional datasets.

3.3.3 Evaluation Measures

For the evaluation of the experiments, we use two of the most well-known classification measures (BALDI *et al.*, 2000; DÉSIR *et al.*, 2013; WANG *et al.*, 2018): Area Under the ROC Curve (ROC AUC) and Matthews Correlation Coefficient (MCC).

MCC

Matthews Correlation Coefficient (MCC) (MATTHEWS, 1975) is a discrete version of Pearson's correlation coefficient (KIRCH, 2008) between the ground truth and the predicted labels. MCC ranges from -1 (all predictions are wrong) to +1 (perfect classification). Null values indicate that either predictions are completely random or the classifier always predicts only one of the classes. MCC can also be calculated from the contingency table as:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (3.1)$$

where True Positive (TP) is the number of true outliers correctly classified, True Negative (TN) the number of true inliers correctly classified, False Negative (FN) the number of true outliers that have been misclassified and False Positive (FP) the number of true inliers misclassified.

The fact that the correlation coefficient takes into account all the four elements from the contingency table (TP, TN, FP, FN) makes it offers a more robust evaluation, suitable even for problems with classes of very different sizes. Most of the other classification measures only take into account the imbalanced problem by defining the positive class (*e.g.* recall, precision, and F_1 score). The advantage of not having to define the positive class is especially important to us because in experiment Type I (Figure 6a) the inlier class tends to be rarer compared to the outlier class, whereas in experiments Type II (Figure 6b) and Type III (Figure 6c) it is the other way around.

ROC AUC

In one-class classification and unsupervised outlier detection, the output of the algorithms is usually continuous, the numbers TP, TN, FP and FN depend on how the threshold is selected. Generally, there is a trade-off between the amount of false positives and the amount of false negatives produced by the algorithm. Receiver Operating Characteristic (ROC) curve plots, for all possible threshold choices, the true positive rate $\left(\frac{TP}{TP+FN}\right)$ versus the false positive rate $\left(\frac{FP}{FP+TN}\right)$. Since the false positive rate is normalized by the number of false positives, and the true positive rate is normalized by the number of true positives, ROC inherently adjusts for the imbalance of class sizes (CAMPOS *et al.*, 2016).

A ROC curve can be summarized in a single value by computing the area under this curve (ROC AUC). The ROC AUC value ranges between 0 and 1. A perfect ranking of the instances (in which all outliers are ranked ahead of any inliers) result in a ROC AUC value of 1, whereas an inverted perfect ranking produce a value approaching 0. A random ranking of the instances result in a ROC AUC value close to 0.5.

For multi-class datasets, the ROC AUC of the different one-class datasets resulting from different labeling of the datasets can be aggregated using the weighted ROC AUC (FAWCETT, 2004):

$$\text{W. ROC AUC} = \sum_{i=1}^m p(i) \times \text{ROC AUC}_i \quad (3.2)$$

where m is the number of classes in the dataset used as inlier, $\text{ROC AUC}(i)$ is the classifier's ROC AUC score for the dataset with the class i as inlier, and $p(i)$ represents the fraction of instances belonging to class i among all the instances in the original dataset that belong to classes used as inliers.

ROC AUC vs. MCC

While ROC AUC takes the entire ranking into account, without the need to establish a threshold, MCC evaluates the classifier performance to that threshold selected by the classifier. A high ROC AUC score indicates only that, in the overall ranking, outliers are more likely to be

ranked ahead of inliers. For the boundary methods, for example, it only means that the inliers are closer to the boundary descriptor than the outliers; it does not necessarily mean that the boundary descriptor is accurately classifying the instances. Figure 7 shows an example of a boundary descriptor that achieves ROC AUC = 1 but misclassifies a large fraction of inliers. Although not trivial for some methods (*e.g.* boundary methods), this problem can be solved by applying an offset in the threshold set by the classifier. MCC, on the other hand, evaluates the performance of the classifier based on only one threshold setting. Incorrectly defining such a threshold may lead to a poor evaluation, even though it may be fixed by applying a different threshold. Therefore, we argue that one cannot rely solely on ROC AUC or MCC scores in judging the quality of a method; rather, ROC AUC and MCC complement each other, as they reveal different aspects of the problem, both of which are relevant in practice.

Figure 7 – Illustrative examples of a boundary descriptors in red. The boundary descriptor achieves a high ROC AUC, albeit a low MCC.

Friedman test

Friedman test (FRIEDMAN, 1937) is a non-parametric statistical test that ranks the algorithms for each dataset, such that the best performing algorithm gets rank of 1, the second best rank 2, and so forth. The null hypothesis states that the average ranks of the compared algorithms are not significantly different. The Friedman statistic

$$\chi^2_{(c-1)} = \frac{12m}{c(c+1)} \left(\sum_{j=1}^c R_j^2 - \frac{c(c+1)^2}{4} \right) \quad (3.3)$$

is distributed according to $\chi^2_{(c-1)}$ with $c - 1$ degrees of freedom, when m and c are big enough (as a rule of thumb, $m > 10$ and $c > 5$), where c is the number of classifiers and m the number of datasets. R_j is the average rank of classifier j among all datasets. For m or c small, the approximation to chi-square becomes poor and the p -value should be obtained from tables specially prepared for the Friedman test (DEMŠAR, 2006). If the null-hypothesis is rejected, *i.e.* the p -value is significant, one can proceed with a post-hoc test.

Neményi test

The post-hoc Neményi test (NEMÉNYI, 1963) is a pairwise test used to check whether there is a significant difference in algorithm performances. The performance of two algorithms is significantly different when the difference between their average ranks is greater than or equal to the critical difference:

$$CD = q_{\alpha} \sqrt{\frac{c(c+1)}{6m}} \quad (3.4)$$

where q_{α} is the critical tabulated value for the test at the specified significance level (NEMÉNYI, 1963). Note that we can decrease the critical difference to claim statistical difference by increasing the number of datasets m or decreasing the number of algorithms c .

3.3.4 Performance Evaluation

In order to evaluate a method's performance on a one-class dataset, we perform the following procedure: First, we split the dataset into 2 subsets, one containing 20% and the other containing 80% of the data. In the subset with 80% of the data we apply a 10-fold cross-validation procedure to optimize the parameters of the methods with respect to ROC AUC.

The parameters of the methods were optimized in the following ranges: $k = 1, 2, \dots, 50$ for LOF, KNN_{global} and KNN_{local} ; $M_{clSize} = M_{pts} = 1, 2, \dots, 50$ for GLOSH; No. hidden units = 2, 5, 7, 10, 12, 15, 17, 20, 22, 25 for Auto-Encoder¹; ε and $h = 0.001$ to 50 (discretized logarithmically in 25 different values) for Gaussian Density and for the Gaussian kernel used in SVDD; α, s, Λ , and $v = 0.1, 0.2, \dots, 1.0$ for LOCI, LP, PW and SVDD respectively.

After parameter optimization, the subset containing 20% of the data (test set) is used to measure the performances of the algorithms (trained with the optimal parameter values from the 10-fold cross-validation). In order to get more reliable results, this procedure is repeated 30 times, and the resulting ROC AUC values are aggregated and reported. In addition to ROC AUC values, we also report the MCC measure for the classification of results obtained on the test sets.

For the sake of comparison with the results reported by Janssens, Flesch and Postma (2009), we also compute the Weighted ROC AUC measure used in their work, which gives more weight to results obtained from experiments involving larger inlier classes. It is worth remarking, however, that this approach may be questionable, since it is well known that ROC curves already inherently adjust for the imbalance of class sizes.

¹ Due to the high computational demand of Auto-Encoder, a time limit of 1000s was imposed to the network convergence, which gives a maximum amount of time equal to 30 repetitions $\times$ 10 folds $\times$ 10 parameters $\times$ 1000s $\approx$ 35 days per experiment involving a single dataset.

3.4 Results

Figure 8 shows the average ranking of the 5 methods compared in (JANSSENS; FLESCHE; POSTMA, 2009) over all Type I experiments w.r.t. weighted ROC AUC using only the real datasets. This figure summarizes our attempt at reproducing Janssens, Flesch and Postma (2009)’s results, which were restricted to this particular setup only. The width of the upper bar (CD) indicates the critical difference of the well-known Friedman/Neményi statistical test at significance level $q_\alpha = 0.05$. The bars within the plots indicate subsets of methods for which there is no significant difference according to the test. The analogous figure in (JANSSENS; FLESCHE; POSTMA, 2009) shows two subsets of methods: (1) the top performers SVDD, LOF, and KNN_{local} (with SVDD and LOF having exactly the same average rank), and (2) a group consisting of PW and LOCI, clearly separated from (1), with much lower performance. Our result does not agree with Janssens, Flesch and Postma (2009) in several respects: first, LOF and SVDD are not tied, second, there are no longer two clearly separated groups, and third, KNN_{local} is the worst performer rather than one of the best.

Figure 8 – Average ranking of the methods compared in (JANSSENS; FLESCHE; POSTMA, 2009) over all Type I experiments w.r.t. weighted ROC AUC.

In the following, we describe the results according to our experimental setup described above. Detailed numbers for all experiments are given in tables in Appendix A. The overall ranking results are summarized by the critical difference diagrams (DEMŠAR, 2006) presented throughout the remainder of this chapter.

Unimodal

Figures 9a and 9b show the average rankings of the algorithms for the experiment Type I on the synthetic datasets. In this scenario of unimodal inlier class, which is the most common in one-class classification literature, the algorithms KNN_{global} , SVDD, and PW are the top performers with respect to ROC AUC. However, SVDD and PW significantly drop position when evaluated with respect to MCC. This suggests that in the outlier scoring produced by these algorithms, the scores of the true outliers tend to be higher than those of inliers, overall. However, the algorithms fail to set the classify threshold, resulting in a high rejection of inliers

or acceptance of outliers, as discussed in section 3.3.3. It worth to notice, however, that there is no statistical significance between Gaussian, $\text{KNN}_{\text{global}}$, GLOSH, SVDD, and ABOD, and they all perform well on both measures. The average ROC AUC was tied in 0.999 for all methods, and the average MCC were 0.944, 0.935, 0.934, 0.933, and 0.926, respectively (see the tables in Appendix A for detailed values).

Figure 9 – Average rankings of the methods over all Type I experiments.
Unimodal — Synthetic datasets

Multi-modal

In the experiments Type II performed on the synthetic datasets, which correspond to application scenarios with possibly multi-modal inlier classes, $\text{KNN}_{\text{global}}$, SVDD and PW are again the top-performs with respect to ROC AUC (Figure 10a). In particular, the relative performance of SVDD and PW drops once more for MCC, as it also does in Type I experiments. When considering the statistical significance, $\text{KNN}_{\text{global}}$ and LOF are the only two methods among the best performers w.r.t. ROC AUC and MCC. The average ROC AUC was 0.995 for both, and the average MCC were 0.828 and 0.788, respectively. When comparing Type I experiments in Figure 9 and Type II experiments in Figure 10, a noticeable difference can be observed with respect to Gaussian: while it was among the top performers in Type I experiments, its relative performance drops sharply in Type II. The absolute performance of Gaussian indeed drops from Type I to Type II experiments, from 0.99 to 0.84 for ROC AUC and from 0.94 to 0.26 for MCC, which is expected as Gaussian presumes a unimodal inlier class model that fits best the data as arranged in Type I experiments. But this alone does not fully explain the drop of Gaussian in terms of relative performance. What also explains it is that other methods, particularly local density-based methods like GLOSH, LOF and even $\text{KNN}_{\text{local}}$, perform better in Type II experiments.

Local outliers

In the experiments Type III performed on the synthetic datasets, which correspond to application scenarios with possibly local outliers, LOF, GLOSH and SVDD are the top-performs

Figure 10 – Average rankings of the methods over all Type II experiments.
Multi-modal — Synthetic datasets

with respect to ROC AUC (Figure 11a). Once again, SVDD performance drops with respect to MCC. When considering the statistical significance for ROC AUC and MCC, LOF and GLOSH are the two best, with no statistical difference. The average ROC AUC were 0.955 and 0.949, and the average MCC were 0.412 and 0.295, respectively. In this scenario, we can draw the same conclusions as in the experiment Type II regarding Gaussian and local density-based methods. However, in the local outlier detection scenario, the performance of local density-based methods is even more prominent and we can see a clear superiority.

Figure 11 – Average rankings of the methods over all Type III experiments.
Local outliers — Synthetic datasets

Small sample size

In the experiments Type IV performed on the synthetic datasets, we use only the variants of the synthetic datasets with less than 20 instances, in order to evaluate the behavior of the methods in datasets with small sample size. When comparing Type I experiments in Figure 9 and Type IV experiments in Figure 12, we can notice, in particular, the decrease in Gaussian and SVDD performances. The decrease in Gaussian performance is expected as the distribution parameters (mean and covariance matrix) are not known and therefore calculated from the data. Small sample size leads to a poor estimation. In the case of SVDD, the decrease in the performance is related to the representativeness of the sample. Although SVDD can work well in a small sample size, the sample needs to be representative. In a small sample size, it is unlikely

that the split in the dataset for training and testing provides representative instances in the training set just by chance.

Figure 12 – Average rankings of the methods over all Type IV experiments.
Small sample size — Synthetic datasets

Large sample size

In the experiments Type V performed on the synthetic datasets, we use only the variants of the synthetic datasets with more than 20 instances, in order to evaluate the behavior of the methods in datasets with large sample size. When comparing Type I experiments in Figure 9 and Type V experiments in Figure 13, we can reaffirm the conclusions of Type IV experiments. In a large sample size, Gaussian and SVDD performances increase, especially Gaussian, which is now the best performer for both measures.

Figure 13 – Average rankings of the methods over all Type V experiments.
Large sample size — Synthetic datasets

High-dimensional

In the experiments Type VI, we use the high-dimensional collection of synthetic datasets (ZIMEK *et al.*, 2013; ZIMEK; CAMPELLO; SANDER, 2014), in order to evaluate the behavior of the methods in high dimensionality. In these high-dimensional datasets, the clusters are well separated, such that most methods perform well w.r.t. ROC AUC. LP is the only method that fails to achieve ROC AUC = 1. However, it is worth to remember that this method works in

the similarity space, and we define the Euclidean distance as the similarity measure. The use of a robust measure for high dimensionality, such as the cosine measure (TAN; STEINBACH; KUMAR, 2005), might improve the results for this method. For MCC, ABOD is the top-perform, which could be expected as the method was designed specifically to work in high dimensionality, using a score based on the cosine measure.

Figure 14 – Average rankings of the methods over all Type VI experiments.
High dimensional — Synthetic datasets

Real datasets

Figure 15 shows the average rankings of the methods over all Type I experiments with respect to ROC AUC and MCC on real datasets. When looking at ROC AUC, one can see SVDD, Gaussian, and kNN_{global} , in this order, at the top. The average ROC AUC for SVDD, Gaussian, and kNN_{global} were 0.78, 0.78, and 0.76, respectively. When looking at MCC the overall picture is similar, but MCC tells us that SVDD is no longer the top performer as it is now outperformed by Gaussian. In contrast to the synthetic datasets, ABOD is no longer among the top-performs; instead, LOF now is one of the top-performs.

Figure 15 – Average rankings of the methods over all Type I experiments.
Unimodal — Real datasets

Overall, based on the types of experiments, we conclude that: (i) in agreement with (JANSSENS; FLESCHE; POSTMA, 2009), SVDD is a top performer, especially with respect to ROC AUC; (ii) kNN_{global} , however, might be a preferable choice, since it is consistently a

top performer, yet it is much simpler than SVDD; this method was not included in the study in (JANSSENS; FLESCHE; POSTMA, 2009); and (iii) in contrast to (JANSSENS; FLESCHE; POSTMA, 2009), LOF does not perform as strongly as SVDD, and $\text{kNN}_{\text{local}}$ is not among the top performers, but consistently among the worst.

3.5 Chapter Remarks

In this study, we provided a comprehensive comparison of one-class classification algorithms and unsupervised outlier detection methods extended to the one-class classification scenario. These methods were evaluated over a number of datasets, measuring performance with respect to ROC AUC and MCC in different experimental settings. The most important conclusion is that SVDD and $\text{kNN}_{\text{global}}$ are the top choices for one-class classification, while we do not recommend $\text{kNN}_{\text{local}}$. This is in contrast to the previous comparison study by Janssens, Flesch and Postma (2009), which did not include $\text{kNN}_{\text{global}}$ and reported $\text{kNN}_{\text{local}}$ as a top performer. In addition, we could not confirm the top performance of LOF reported in (JANSSENS; FLESCHE; POSTMA, 2009), but only that of SVDD.

Interestingly, SVDD is recognized in the literature for performing better on real datasets than on synthetic datasets (TAX, 2001). Although SVDD also performed well on synthetic datasets, our experiments can confirm this conclusion, in particular for MCC. The fact that the relative SVDD performance consistently drops for MCC may be related to the parameter ν , which allows SVDD to reject a fraction of the training set, resulting in decision boundaries similar to the one in Figure 7. This feature seems to be interesting for real datasets, as they have more complex characteristics such as noise and overlap. For more controlled scenarios, this may result in a performance loss, such as in the experiments in high dimensional datasets (Figure 14), the loss in SVDD performance for this experiment is more related to the fraction of training instances rejected than the unsuitability of SVDD for high dimensional datasets.

Regarding more specific application scenarios, we also can draw the following conclusions: If a unimodal distribution is guaranteed for the inlier class and we have a reasonable sample size, the Gaussian model is an interesting alternative, even if the data distribution is unknown, as in the experiment on real datasets. The local density-based methods perform better in the multi-modal scenario than in the unimodal; nevertheless, they do not outperform $\text{KNN}_{\text{global}}$. For local outliers, the local density-based methods are the obvious choice, in particular LOF and GLOSH. For high-dimensional datasets, ABOD performs the best. However, it seems interesting to adapt other methods, *e.g.* $\text{KNN}_{\text{global}}$ and LP, to use the cosine measure.

In addition to the above, we described an adaptation of a recent outlier detection method — called GLOSH (CAMPELLO *et al.*, 2015) — to the one-class classification problem. We also discussed basic principles of one-class classification that should not be violated when adapting

unsupervised outlier detection methods to this task, and how to assure that such principles are not violated.

The procedure for model selection (parameters) and evaluation of results performed in this chapter were based on the use of previously labeled datasets. The labels were not used by the algorithms, but rather to assess their results (ROC AUC and MCC) and select the best model/parameter (cross-validation). This procedure is referred to as external evaluation or validation, because it makes use of external information, *i.e.*, information that the algorithm does not have access (labels). Although the use of external information makes sense when comparing performances of algorithms in controlled experiments, such as those in this chapter, it is not feasible in practical applications for semi-supervised and unsupervised learning. In real-world applications where a fully labeled dataset required by external evaluation measures is not available, this problem is usually tackled by using internal evaluation, which is based solely on the data and the assessed solutions themselves. In the following chapters, we discuss the problem of internal evaluation for one-class classification (chapter 4) and unsupervised outlier detection (chapter 5).

MODEL SELECTION FOR ONE-CLASS CLASSIFICATION

Since in one-class classification problem only examples of the inlier class are available representatively, it is hard to decide how tightly the model should fit around the data. The one-class classification models typically contain parameters which determine how tightly the model should fit around the data. The tighter the model, less likely it is to flag outliers as inlier. Models too tight, however, come at the price of overfitting. Although the setting of these parameters gives the possibility of making more flexible and better fitting models, the choice for these parameters is not intuitive and the decision is often left to the user.

In this chapter, we discuss this problem and provide solutions for model selection in one-class classification. More specifically, we propose a new model selection method. The method, which is based on the perturbation of the data, is evaluated in practical experiments involving model selection and compared against other methods in the same rigorous experimental setup as chapter 3.

4.1 Introduction

In one-class classification, the instances from one of the classes (inlier class) have to be discriminated from all other possible instances that do not belong to that class. In this context, the instances not belonging to the inlier class are called outliers. The general goal of this task is to find a model that only accepts instances belonging to the inlier class. However, unlike the traditional classification problem (DUDA; HART; STORK, 2000; TAN; STEINBACH; KUMAR, 2005), in one-class classification, it is usually assumed that we are only provided with examples from the inlier class because outliers are either nonexistent or very rare and insufficient for the generalization of a classifier.

Since in one-class classification problems only one of the classes is available representatively, the model is supported only by the instances from that class. Therefore, it is hard to decide how tightly the model should fit around the data. The one-class classification models typically contain parameters which determine how tightly the model should fit around the data, *i.e.*, how complex the model should be. The more complex the model, the more flexible and tighter it fits the inlier class. Models too complex, however, come at the price of overfitting, where the model mainly adapts individually to each instance from the inlier class, which results in high variance. On the other hand, models too simple lead to underfitting, which results in a high bias.

Although the setting of these parameters gives the possibility of making more flexible and better fitting models, the choice for these parameters is not intuitive and the decision is often left to the user. The selection for these parameters, however, has a great influence on the final performance of the models. In one-class classification, these parameters determine the trade-off between the error on the inlier class and the error on the outlier class, such that poor parameter choices lead to poor classification performances.

In the literature, the most common procedure for model selection (algorithm, parametrization) in one-class classification is based on the use of previously labeled datasets, in which, not only the inliers are previously known, but also the outliers. In this scenario, referred to as external evaluation or validation, the labeled outliers are not used by the algorithms, but rather to assess their results only, from which the best performing algorithms/parameters can be identified. Specifically, the labeling or ranking produced by the algorithm is compared with the correct and previously known labeling (ground truth). The quantitative result of this comparison is taken as the quality measure using, for example, MCC or ROC AUC curve (BALDI *et al.*, 2000; CAMPOS *et al.*, 2016). However, the availability of a fully labeled dataset required by external evaluation measures is not consistent with the premises of semi-supervised learning, and the commonly practiced external evaluation for one-class classification makes sense only when comparing performances of algorithms in controlled experiments.

Some works in the literature have been dealing with the problem of the absence of outliers for performing model selection. These works fall mainly into two categories: Methods designed for one specific algorithm (DUIN, 1976; TAX; MÜLLER, 2004; XIAO; WANG; XU, 2015; GHAFOORI *et al.*, 2016), which restricts to the selection of the best parameter for that specific algorithm (parameter tuning); and methods designed for any kind of algorithm (TAX; DUIN, 2001b; WANG *et al.*, 2018), which is suitable for the selection of both the best algorithm and parametrization. This chapter focus on the latter category of methods. These methods are internal in the sense that they only use internal information, and relative by pointing out which model is better in relative terms. In this chapter, we propose to introduce perturbation on the data as a new way to guide the model selection using only internal information. We evaluate this new approach for model selection in practical experiments to select the best algorithm and its parametrization and compare it to other model selection methods in the same rigorous

experimental setup as chapter 3.

The remainder of this chapter is organized as follows. In section 4.2, we introduce the model selection methods for one-class classification. In section 4.3, we describe the experiments and discuss the results. Finally, we summarize and conclude the chapter in section 4.4.

4.2 Model Selection Methods for One-Class Classification

Model selection methods for one-class classification usually rely on the classification error to guide the search for the best model (parameterization/algorithm). These methods search for the best model, such that the classification error is minimized. In most one-class classification problems, we are not only interested in models that have a small overall error, but also in what type of instances are misclassified. Due to the inherited imbalance of the outlier class, it is easy to imagine examples where a model has a small overall error, but still misclassifying a high ratio of outliers. In one-class classification problems, where misclassifying outliers as inliers is especially dangerous, *e.g.* fraud detection, we are also interested in the minimization of the error in the outlier class. Therefore, the error of a one-class classification model, in general, can be expressed as:

$$\Lambda(\varepsilon_i, \varepsilon_o) = \alpha \varepsilon_i + (1 - \alpha) \varepsilon_o \quad (4.1)$$

where ε_i is the inlier rejection rate, *i.e.*, the rate of inliers classified as outliers, and ε_o is the outlier acceptance rate, the rate of outliers classified as belonging to the inlier class. α controls the trade-off between the importance of outlier acceptance (ε_o) and inlier rejection (ε_i). For $\alpha = 0.5$ both errors are equally treated. Having estimates of ε_i and ε_o , we can estimate classification error for the models to point out which one is better in relative terms. The problem, however, is that in one-class classification only instances of the inlier class are available during the training phase. Therefore, only ε_i can be estimated on the training set using traditional classification techniques, such as cross-validation and hold-out (HAN; KAMBER; PEI, 2011; TAN; STEINBACH; KUMAR, 2005). The expected error on the outlier class (ε_o) only can be estimated by making additional assumptions. In the next section, we present two methods capable of estimating the classification error and, therefore, performing model selection for one-class classification. In addition, we also propose a new method for model selection based on the perturbation of the data.

4.2.1 Uniform Object Generation

The most common assumption concerning the distribution of the outliers is that the outlier class is uniformly distributed in the input space (TAX; DUIN, 2001b). In Tax and Duin (2001b), the authors rely on this assumption to fit the smallest enclosing hypersphere around the training set, computed by a linear SVDD algorithm (TAX; DUIN, 2004), and generate Q uniformly distributed outliers inside it. The ratio of generated outliers that are classified as inliers is the estimation of the error on the outlier class (ϵ_o). The idea behind this approach is that when the volume covered by the one-class classification model is minimized, the chance of accepting outliers is also minimized. The problem is that in high dimensional feature spaces, as pointed out by the authors themselves, this procedure becomes very computationally inefficient as it requires a large number of artificially generated outliers to minimally cover a high dimensional space. For 100,000 uniformly distributed objects, this method is accurate up to 15-20 dimensions, depending on the data distribution (JUSZCZAK, 2006). In order to estimate the error on the inlier class (ϵ_i), the authors propose to use cross-validation, since the labels for inlier class are available.

4.2.2 Self-Adaptive Data Shifting

More recently, Wang *et al.* (2018) proposed the Self-adaptive Data Shifting (SDS) method. This method relies on an Edge Pattern Detection (EPD) algorithm to find the edges of the data. After the EDP algorithm finds the edge patterns, for a given neighborhood and a threshold, these instances are used to generate pseudo outliers. The pseudo outliers are generated by shifting these instances along the direction of the negative estimated data density gradient. In addition to the pseudo outliers, this method also generates pseudo inliers by shifting each instance of the dataset along the positive direction of the data density gradient. After generating pseudo instances for both classes, the one-class classification model uses the original data for training and the generated data as the testing set to estimate ϵ_i and ϵ_o . The main weakness of this method relies on the EDP algorithm, in high dimensional spaces, the datasets tend to become sparse, and the EDP algorithm begins to see every instance as an edge pattern.

4.2.3 Data Perturbation

The process of data perturbation consists of the change of the attribute values of each instance by adding a noise component of some small, randomized, amount. At first sight, it may seem counterintuitive that, while much effort is spent to reduce or eliminate noise in datasets, adding noise to data should actually help in the data mining process. However, Rand (1971) and Zimek, Campello and Sander (2014), for example, already benefited from this addition of controlled noise. Rand (1971) proposed to use the data perturbation in the way that it is mainly

used in data mining nowadays, in the evaluation of clustering results by assessing the clustering stability by comparing the original partitions with the partitions produced after the perturbation.

More recently, [Zimek, Campello and Sander \(2014\)](#) applied the data perturbation to introduce diversity into ensembles for outlier detection. The data perturbation is applied to generate different instances of the dataset. In these different instances, the authors expect to observe the algorithms making ranking inversions more frequently for the instances at the border of the cluster/class, while the clear inliers and top outliers would remain more stable, respectively, at the bottom and top of the rankings. When combined the ranks of the several versions of the dataset produced through the perturbation, the authors expect to obtain an improved result, especially for borderline cases.

Similarly to [Rand \(1971\)](#), we propose to use data perturbation in the evaluation, but instead of evaluating the stability of partitions, we use to evaluate the performance of the model on the outlier class. We borrow some of the [Zimek, Campello and Sander \(2014\)](#) ideas, especially regarding the borderline cases, to formulate our hypothesis. The basic idea is that when the perturbation is applied to the data, most of the borderline instances would be affected enough to become outliers. The tighter the fit of the model to the data, the higher the number of instances close to the decision boundary. Therefore, these models with a tighter fit should present a larger number of instances that become outliers after the perturbation. Based on that, we follow the same procedure of [Zimek, Campello and Sander \(2014\)](#) to generate different versions of the dataset using data perturbation. In these different versions, we expect that the models most fitted to the data, according to our hypothesis the models with the lowest errors on the outlier class (ϵ_o), to have the highest rejection rates of such perturbed instances.

The example of Figure 16 illustrates our intuition. In Figure 16a, 16b and 16c, different models are applied to the original data. In the model of Figure 16a, only a small fraction of the instances are close to the decision boundary. When we apply this model to the perturbed data (Figure 16d), no instance is rejected. On the other hand, the model of Figure 16b fits better to the data and, thus, a higher fraction of the instances are close to the decision boundary. Therefore, when we apply this model on the perturbed data (Figure 16e), a higher rate of instances are rejected.

Notice, however, that our hypothesis only concerns with the error on the outlier class, we make no assumptions about the error on the inlier class. The model of Figure 16c is the most complex among the three. When we apply this model to the perturbed data (Figure 16f), it is the one with the highest rejection rate. Following our hypothesis, it is the one that presents the smallest error on the outlier class (ϵ_o). In one-class classification, however, the error has two components, as shown in Equation (4.1). Therefore, even though the model has a small error on the outlier class, ϵ_i has also to be computed (*e.g.* using cross-validation) and combined with the rejection rate in the perturbed data. Note also that using only ϵ_i to evaluate the results will not lead to good results either, the model of Figure 16a achieves a low ϵ_i error through

cross-validation and it probably has a high ε_o error.

Figure 16 – Illustrative example.

Noise generation

The process of data perturbation consists of the change of the attribute values of each instance by adding a noise component of some small, randomized, amount. The magnitude of the added noise component is very important. A too small amount will make almost all perturbed data to remain inside of the decision boundary, and only models extremely overfitted would benefit from the evaluation. On the other hand, an amount too large will make all the perturbed data to fall far outside from the decision boundary, and thus, no model would benefit from the evaluation. Therefore, we implement the perturbation by adding to each instance an attribute-wise noise component sampled from $\mathcal{N}(0, \sigma)$ in each attribute. In order to control the magnitude of the noise component, we select σ based on the fraction of perturbed instances rejected by a linear SVDD (TAX; DUIN, 2004). We start by using a small fraction of σ and gradually increase until the linear SVDD rejects 50% of the data. The idea of rejecting 50% of the perturbed data is that we need instances on both sides of the decision boundary. Therefore, we choose to distribute the perturbed instances evenly inside and outside the decision boundary. We use this fraction of σ to apply the perturbation and generate different instances of the dataset. The outlier acceptance rates of the model over the different instances of the dataset are averaged and used as our estimate of ε_o . Finally, we evaluate the error of the model by combining our estimate of ε_o with ε_i (estimate, e.g., using cross-validation) in Equation (4.1).

4.2.4 Complexity

The asymptotic computational complexity of our method depends on the complexity of the one-class classifier. The computational complexity of the one-class classification algorithms can be viewed as occurring at two distinct times, the training time $\mathcal{O}(f_{tr}(N))$ and the testing (or consultation) time $\mathcal{O}(f_{te}(M))$, as a function of the training set size (N) and the testing set size (M), respectively. In general, the algorithms can be divided into two groups, eager learning and lazy learning (WEBB, 2011). In eager learning algorithms, the majority of computation occurs at training time. The SVDD (TAX; DUIN, 2004) is an example of an eager learning algorithm, which has a training complexity of $\mathcal{O}(N^3)$, but once the model is built, it classifies new instances in linear time ($\mathcal{O}(M)$). In contrast to eager learning, lazy learning spends more time on testing time. Examples of algorithms of this category are the neighborhood-based algorithms (BREUNIG *et al.*, 2000; RAMASWAMY; RASTOGI; SHIM, 2000), these algorithms spend no time in the training phase, and $\mathcal{O}(N \cdot M)$ to classify new instances.

Once the classifier is trained, we can divide the process of error evaluation into three phases: the generation of objects that we use to estimate ε_i and/or ε_o , the estimate of ε_i and the estimate of ε_o . The phase of objects generation of our method involves the training of a linear SVDD, which can be efficiently computed in linear time ($\mathcal{O}(N)$) (JOACHIMS, 2006; ERFANI *et al.*, 2015).¹ The estimate of ε_i takes $\mathcal{O}\left(k \cdot \left(f_{tr}\left(\frac{(k-1) \cdot N}{k}\right) + f_{te}\left(\frac{N}{k}\right)\right)\right)$ using k -fold cross-validation. The estimate of ε_o depends on the number of different instances of the dataset generated using perturbation (m). In these different instances, the classifier has to test each object, which takes $\mathcal{O}(m \cdot f_{te}(N))$.

In Table 1, we provide the complexity of other methods in the literature. SDS (WANG *et al.*, 2018) has some advantage over our method in estimating ε_i and ε_o , however, we have a clear advantage in the phase of objects generation. The complexity of uniform objects (TAX; DUIN, 2001b) is the one that most resembles ours, differing only in the estimate of ε_o . Uniform objects has the advantage of not depending on the size of the dataset, which is good for large datasets. However, as pointed out by the authors themselves, the number of uniform objects (Q) that are required grows fast with the increase of the dimensionality, which makes it very expensive for high dimensional datasets.

	Gen.	ε_i	ε_o
Perturbation	$\mathcal{O}(N)$	$\mathcal{O}\left(k \cdot \left(f_{tr}\left(\frac{(k-1) \cdot N}{k}\right) + f_{te}\left(\frac{N}{k}\right)\right)\right)$	$\mathcal{O}(m \cdot f_{te}(N))$
Uniform Objects	$\mathcal{O}(N)$	$\mathcal{O}\left(k \cdot \left(f_{tr}\left(\frac{(k-1) \cdot N}{k}\right) + f_{te}\left(\frac{N}{k}\right)\right)\right)$	$\mathcal{O}(f_{te}(Q))$
SDS	$\mathcal{O}(N \log N)$	$\mathcal{O}(f_{te}(N))$	$\mathcal{O}(f_{te}(N))$

Table 1 – Computational complexity

¹ Note that only the linear SVDD can be computed in $\mathcal{O}(N)$. The use of kernel makes SVM computation becomes $\mathcal{O}(N^3)$.

4.3 Experiments

We compare and evaluate the 3 model selection methods presented in section 4.2: Perturbation, Uniform Objects (TAX; DUIN, 2001b), and SDS (WANG *et al.*, 2018). For Perturbation, we generate 10 instances of perturbed datasets to give more stability to the results. For Uniform Objects, we generate $Q = 100,000$ uniformly distributed objects as this number was also used by the authors (TAX; DUIN, 2001b) as the maximum number of uniformly generated objects. For SDS, the neighborhood size and the threshold used by the EDP algorithm was fixed as suggested by the authors as $\lceil 5 \log_{10} N \rceil$ and 0.1, respectively. We use 10 folds cross-validation to estimate ε_i in Uniform Objects and Perturbation. After the estimation of ε_i and ε_o , we estimate the classification error using $\alpha = 0.5$ to give equal weights for the inlier and outlier class. We use code from the repository available at <http://prlab.tudelft.nl/users/david-tax/> (TAX, 2018) for Uniform Objects. For SDS, we use our implementation based on the one made available at <https://www.ipd.kit.edu/ocal/> (TRITTENBACH; ENGLHARDT; BÖHM, 2018), we only translate their implementation to MATLAB to make it compatible with the remaining of our code. In the case of Perturbation, we used our own implementation.

We use the same experimental setup described in the previous chapter in our experiments, we differ only in the way we select the models. Instead of using outliers to select the models, we proceed as follow: First, we split the dataset into 2 subsets, one containing 20% and the other containing 80% of the data. As the model selection methods can not use the outliers in any way, the outliers are included only in the subset containing 20% of the data (test set) (JUSZCZAK *et al.*, 2009). In the subset with 80% of the data, we trained the algorithms SVDD, KNN_{global} , and LOF in the same range of parameter used in chapter 3.² Each of the model selection methods selects the best model (algorithm and parameter) based on their own estimation of the classification error on the training set, as discussed in section 4.2. After the model selection, the subset containing 20% of the data (test set) is used to measure the performances of the selected models using ROC AUC and MCC.

In the following, we describe the results according to our experimental setup described in the previous chapter and supplemented by the paragraphs above. Detailed numbers for all experiments are given in tables in the Appendix B. The overall ranking results are summarized by the critical difference diagrams (DEMŠAR, 2006) presented throughout the remainder of this chapter.

² We selected these three algorithms because of their overall performance in the different scenarios evaluated in the last chapter.

4.3.1 Results

Figure 17 shows the average rankings of the algorithms for the experiment Type I on the synthetic datasets. In this scenario of unimodal inlier class, which is the most common in one-class classification literature, the method based on perturbation selects the best models with respect to ROC AUC and MCC. For ROC AUC, however, there is no statistical difference between the methods, and they all perform well. The average ROC AUC for the methods were 0.999. For MCC, the method based on perturbation consistently outperforms the others with large statistical significance. The average MCC for Perturbation, Uniform Objects, and SDS were, respectively, 0.909, 0.891 and 0.882 (see the tables in Appendix B for detailed values).

Figure 17 – Average rankings of the methods over all Type I experiments.
Unimodal — Synthetic datasets

Multi-modal

In the experiments Type II performed on the synthetic datasets, which correspond to application scenarios with possibly multi-modal inlier classes, SDS selects the best models with respect to ROC AUC (Figure 18a), although there is no statistical difference. The average ROC AUC for SDS and Uniform Objects were 0.991, while for Perturbation was 0.987. In particular, when comparing Type I experiments in Figure 17a and Type II experiments in Figure 18a, a noticeable difference can be observed in the relative and absolute performance of Perturbation. The drop in its performance is expected due to the perturbation applied to the data being global rather than local. For multi-modal classes with very different sizes/dispersions, when applied the same perturbation for the different modal, the same perturbation for a modal might be too high, while for the other, it might be too low. Notice, however, that such a scenario occurs only in a small fraction of the datasets variants. For ROC AUC, where the other two methods also perform well, this is enough though to drop the relative performance of the Perturbation from the best to the worst model selection method. For MCC, however, where the other methods do not perform as well as Perturbation, it is still the best with statistical significance, even though its absolute performance dropped from 0.909 to 0.839.

Figure 18 – Average rankings of the methods over all Type II experiments.
Multi-modal — Synthetic datasets

Local outliers

In the experiments Type III performed on the synthetic datasets, which correspond to application scenarios with possibly local outliers, SDS selects the best models with respect to ROC AUC (Figure 19a) followed by Perturbation and Uniform Objects, this time with statistical difference between the methods. The average ROC AUC for the methods were 0.920, 0.901, and 0.884. Once again, SDS has the best performance with respect to ROC AUC and significantly drops with respect to MCC. The fact of SDS consistently performs the best with respect to ROC AUC, and the worst with respect to MCC is related to the fact that this method tends to underestimate the error in the inlier class. Remember that, to estimate the error in the inlier class, this method generates pseudo inliers by shifting the training instances in the direction of the density gradient (cluster center). By doing so, the test set becomes much more compact than the training set, favoring models similar to Figure 7, which has high ROC AUC and low MCC.

Figure 19 – Average rankings of the methods over all Type III experiments.
Local outliers — Synthetic datasets

Small sample size

In the experiments Type IV performed on the synthetic datasets, we use only the variants of the synthetic datasets with less than 20 instances, in order to evaluate the behavior of the model selection methods in datasets with small sample size. When comparing Type I experiments in Figure 17 and Type IV experiments in Figure 20, we can notice, in particular, the decrease in

the performances of the models selected by Perturbation. It drops from the first position, with a statistical difference for MCC, to the second position. As Perturbation do not generate new instances, but only perturbed the instances of the dataset, it is expected to not work well in a small sample size. For Uniform Objects, the performance of the selected methods also drops. However, for different reasons, as Uniform Objects generates its own instances. This reason for the drop in the performance for Uniform Objects is related to the fact that the method only generates objects inside of the linear SVDD that encloses the data. As discussed in chapter 3, SVDD does not perform well in small sample size, which results in a poor distribution of the uniform objects.

Figure 20 – Average rankings of the methods over all Type IV experiments.
Small sample size — Synthetic datasets

Large sample size

In the experiments Type V performed on the synthetic datasets, we use only the variants of the synthetic datasets with more than 20 instances, in order to evaluate the behavior of the model selection methods in datasets with large sample size. When comparing Type IV experiments in Figure 20 and Type V experiments in Figure 21, we can notice a clear improvement in the Perturbation and Uniform Objects performances due to the reasons discussed above.

Figure 21 – Average rankings of the methods over all Type V experiments.
Large sample size — Synthetic datasets

High-dimensional

In the experiments Type VI, we use the collection of high-dimensional synthetic datasets (ZIMEK *et al.*, 2013; ZIMEK; CAMPELLO; SANDER, 2014), in order to evaluate the behavior of the model selection methods in high dimensionality. In these high-dimensional datasets, the clusters are well separated, such that all methods achieve ROC AUC = 1. For MCC, Uniform Objects selects the best models. It is in contrast to what is expected, as this method degenerates in high-dimensionality. The reason why Uniform Objects selects the best models is that, for the estimation of the error in the outlier class, a high number of objects is needed to fill the high-dimensional space. As the number of objects generated is not enough, the error in the outlier class tends to zero for every model, and the method relies only on the error of the inlier class. Since the inlier class is well separated from the outlier class (other clusters), there is no risk of outlier acceptance, and relying solely on the inlier class in this scenario, makes the method improve the results. Therefore, the fact of Uniform Objects degenerates in this scenario actually improves its results for this collection of datasets.

Figure 22 – Average rankings of the methods over all Type VI experiments.
High-dimensional — Synthetic datasets

Real datasets

Figure 23 shows the average rankings of the methods over all Type I experiments with respect to ROC AUC and MCC on real datasets. When looking at ROC AUC, one can see SDS, Perturbation, and Uniform Objects, in this order, at the top. This particular order agrees with most of the experiments performed on synthetic datasets. The average ROC AUC for SDS, Perturbation, and Uniform Objects were 0.737, 0.739, and 0.733, respectively. When looking at MCC, the overall picture is also similar to the experiments on synthetic datasets. However, instead of SDS being the worst, it is now the second. It may be related to the same reasons discussed in the previous chapter for SVDD to work well on real datasets. As real datasets are much more complex than synthetic ones, models that reject a fraction of the datasets in trade for a tighter boundary tend to work better. As discussed in the section of local outliers, SDS tends to select this kind of models.

Figure 23 – Average rankings of the methods over all Type I experiments.
Unimodal — Real datasets

4.4 Chapter Remarks

In this chapter, we discussed the problem of model selection in one-class classification. A new model selection method is proposed and compared to other model selection methods over a number of synthetic and real datasets, measuring performance with respect to ROC AUC and MCC in different experimental settings. The most important conclusion is that Perturbation and SDS are the top choices for model selection in one-class classification, while we do not recommend Uniform Objects.

In most general application scenarios, Perturbation might be a preferable choice, since it is consistently a top performer for MCC and still competitive for ROC AUC, while SDS is only consistently a top performer for ROC AUC, for MCC it is consistently the worst. However, we could notice that Perturbation does not perform well in scenarios of multi-modal inlier class with modal of very different sizes/dispersions, and in scenario of small sample size.

Regarding SDS ([WANG *et al.*, 2018](#)), it tends to favor models which reject a fraction of instances in the edge of the clusters/classes, such as SVDD does. As discussed in the previous chapter for SVDD, and confirmed in the experiments on real datasets for SDS, this characteristic seems to be interesting for more complex real-world datasets with noise and overlap.

For Uniform Objects ([TAX; DUIN, 2001b](#)), we could not report a scenario where it is truly a top performer. However, this method is of importance in the literature, as it was the first, and for a long time, the only one capable of performing model selection in the context discussed in this chapter.

INTERNAL EVALUATION OF UNSUPERVISED OUTLIER DETECTION

Although there is a large and growing literature that tackles the unsupervised outlier detection problem, the unsupervised *evaluation* of outlier detection results is still virtually untouched in the literature. The so-called internal evaluation, based solely on the data and the assessed solutions themselves, is required if one wants to statistically validate (in absolute terms) or just compare (in relative terms) the solutions provided by different algorithms or by different parameterizations of a given algorithm in the absence of labeled data. However, in contrast to unsupervised cluster analysis, where indexes for internal evaluation and validation of clustering solutions have been conceived and shown to be very useful, in the outlier detection domain this problem has been notably overlooked.

In this chapter, we discuss this problem and provide a solution for the internal evaluation of outlier detection results. Specifically, we describe an index called IREOS that can evaluate and compare different candidate outlier detection solutions. Initially, the index is designed to evaluate binary solutions only. We then extend IREOS to the general case of non-binary solutions. We also statistically adjust IREOS for chance and extensively evaluate it in several experiments involving different collections of synthetic and real data sets.

5.1 Introduction

Outlier detection techniques can be categorized in different ways. For instance, a common distinction is that between the methods that assign binary labels (“outlier” vs. “inlier” for those observations deemed anomalous vs. normal) and methods that assign a score representing a degree to which an observation is considered to be outlier. Another distinction is that between supervised, semisupervised, and unsupervised outlier detection techniques ([HAN](#); [KAMBER](#);

PEI, 2011). Supervised techniques assume that a set of observed instances labeled as inliers and outliers are available to train a classifier. In the semisupervised scenario, labeled outliers are not available and only previously known inliers can be used in order to obtain a (one class) classification model. When no labeled data are available at all, it is necessary to use unsupervised techniques, which do not assume any prior knowledge about which observations are outliers and which are inliers.

In this chapter, we focus on unsupervised outlier detection scenarios. In general, an outlier in this context can be described as “*an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data*” (BARNETT; LEWIS, 1994). In this context, there is no generally applicable definition of “appearance of inconsistency”; its formalization rather depends on the application scenario and the detection method to be used. A common scenario is to apply some outlier detection method to a database with N observations, labeling a certain subset of n such observations as the n most likely outliers, while labeling the remaining $N - n$ observations as inliers. This is referred to as the *top- n* outlier detection problem (ANGIULLI; FASSETTI, 2009; ANGIULLI; PIZZUTI, 2005; BAY; SCHWABACHER, 2003; GHOTING; PARTHASARATHY; OTEY, 2008; JIN; TUNG; HAN, 2001; KOLLIOS *et al.*, 2003; KRIEGEL; SCHUBERT; ZIMEK, 2008; VU; GOPALKRISHNAN, 2009; ORAIR *et al.*, 2010; RAMASWAMY; RASTOGI; SHIM, 2000), which is far from trivial, especially when dealing with multivariate data following complex, unknown distributions. Without labeled examples, the main complicating factor in this problem is that the notion of “outlierness” is not precisely and generally defined.

The lack of a single, formal, and agreed-upon definition of unsupervised outlier detection is one of the main reasons why a rich variety of detection methods has been developed, from classic parametric statistical methods (BARNETT; LEWIS, 1994; HAWKINS, 1980) to more recent database-oriented approaches conceived to deal with multivariate, possibly large databases. Considering the latter category, a plethora of detection algorithms has emerged in the past 15 years or so. Examples are DB-Outlier (KNORR; NG, 1998; KNORR; NG; TUCAKOV, 2000), kNN Outlier (RAMASWAMY; RASTOGI; SHIM, 2000; ANGIULLI; PIZZUTI, 2005; ANGIULLI; FASSETTI, 2009), LOF (BREUNIG *et al.*, 2000) and its many variants (JIN *et al.*, 2006; KRIEGEL *et al.*, 2009; KRIEGEL *et al.*, 2011; PAPADIMITRIOU *et al.*, 2003; TANG *et al.*, 2002; ZHANG; HUTTER; JIN, 2009), and ABOD (KRIEGEL; SCHUBERT; ZIMEK, 2008), just to mention a few (see, *e.g.*, the works of Schubert, Zimek and Kriegel (2014b) and Campos *et al.* (2016) for discussions of these and many more variants). Each of these algorithms, however, uses its own criterion to judge quantitatively the level of adherence of each observation with the concept of outlier, from a particular perspective. This complicates not only the selection of a particular algorithm and/or the choice of an appropriate configuration of parameters for this algorithm in a practical application, but also the assessment of the *quality* of the solutions obtained, especially in light of the problem of defining a measure of quality that is not tied to the criteria used by the algorithms themselves. These issues are interrelated and refer to the

problems of model selection and assessment (evaluation or validation) of results in unsupervised learning. These problems have been investigated for decades in the area of unsupervised data clustering (JAIN; DUBES, 1988), but are rarely mentioned and are virtually untouched in the area of outlier detection (ZIMEK; CAMPELLO; SANDER, 2013).

In the data clustering domain, the related problems of evaluation and model selection are tackled by using some kind of quantitative index, called validation criterion (JAIN; DUBES, 1988). In practice, when labels are not available, *internal* validation indexes can be used. These indexes are called internal as they do not make use of any external information (such as class labels) in the evaluation of a solution. Instead, internal indexes measure the quality of an obtained clustering solution based only on the solution and the data objects. Most such indexes are also *relative* in the sense that they can be employed to compare different clustering solutions pointing out which one is better in relative terms. Therefore they can also be used for model selection. Internal, relative indexes have been shown to be effective and useful tools for the unsupervised clustering evaluation and model selection tasks — *e.g.* see (HALKIDI; BATISTAKIS; VAZIR-
GIANNIS, 2001; MILLIGAN; COOPER, 1985; VENDRAMIN; CAMPELLO; HRUSCHKA, 2010; VENDRAMIN; JASKOWIAK; CAMPELLO, 2013) and references therein.

The areas of clustering and outlier detection are related to each other and, from a certain perspective, they can even be seen as two sides of the same coin. In fact, when referring to an outlier as “*an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism*”, as in Hawkins’ (HAWKINS, 1980) definition, it is implicitly assumed that there are one or more mechanisms responsible for generating the normal, “unsuspicious” observations. Clusters are possible candidates to model such mechanisms. Surprisingly, although the internal evaluation problem has been extensively studied in data clustering, it has been completely neglected in outlier detection. In this thesis, we step towards bridging this gap by proposing an internal, relative evaluation measure for unsupervised outlier detection. We start from a definition of “outlierness” that is not tied to any particular criterion used by outlier detection algorithms. Rather, it follows the same, common intuition as a multitude of these algorithms and criteria: an outlier is an observation that is to some extent farther away and can therefore be more easily separated from other observations than an inlier. We formulate separability for outlier detection in an objective and principled way, leading to a natural definition of the proposed index.

In summary, we make the following initial contributions:

- We describe the first internal, relative validation measure for evaluation of outlier detection results, IREOS (Internal, Relative Evaluation of Outlier Solutions).
- We describe an improved version of IREOS that is adjusted for chance by removing from the index the theoretical offset that is expected to be observed when evaluating random

solutions. In addition to the adjusted index, we also devise means to return p -values with respect to the null hypothesis of a random solution.

- Since the exact procedure to adjust the index for chance can be computationally demanding for large data sets, we also provide a faster version of the proposed procedure, based on Monte Carlo experiments.

This chapter is an extension of the master’s work of the author of this thesis (MARQUES, 2015; MARQUES *et al.*, 2015), where IREOS has been specifically conceived for, and experimentally evaluated in, the particular scenario of *top-n* (*i.e.*, binary) outlier detection only. In this extension, we make the following additional contributions:

- We extend IREOS to the more general scenario of non-binary outlier solutions, which involves the evaluation of outlier detection *scorings*, which is the type of result produced by most widely used database-oriented algorithms in the literature (CAMPOS *et al.*, 2016). The extended index reduces to the original one when evaluating binary solutions, *i.e.*, the original, *top-n* index is a particular case of the generalized version proposed here.
- We extensively evaluate the extended IREOS index using different collections of synthetic (30) and real (27) data sets, both in controlled experiments as well as in practical experiments of model selection involving a number (13) of different outlier detection algorithms.
- As an additional contribution, we show how IREOS can be used as an auxiliary tool to help characterize the suitability of datasets for outlier detection benchmarking, by assessing the corresponding ground truth labels of these datasets as candidate outlier solutions.

The remainder of this chapter is organized as follows. In section 5.2, we discuss as related work the typical approaches for *external* evaluation and the different requirements and use cases for *internal* evaluation. In section 5.3, we introduce IREOS, discussing requirements and solutions, adjustment for chance, and properties. We experimentally evaluate the proposed index in section 5.4 and conclude the chapter in section 5.5.

5.2 Related Work

In the previous literature, the evaluation of unsupervised outlier detection results has been mostly restricted to making use of labeled data sets to evaluate how algorithms compare to each other, by assessing, in a supervised way, observations previously known to be inliers or outliers according to a particular intuition or semantic (*e.g.*, normal patients versus patients with an uncommon pathology). Note that in this scenario, referred to as *external* evaluation

or validation, the labels are only used to assess the results of otherwise unsupervised outlier detection methods (CAMPOS *et al.*, 2016). For the external evaluation of a top- n outlier detection solution, one is given a data set with n known outliers (ground truth) as well as the observations ranked top- n by the given solution. *Precision-at- n* (prec@ n for short) measures the fraction of the true outliers (*i.e.*, labeled in the ground truth as outlier) among the top- n objects of the given solution (CRASWELL, 2009). If an outlier ranking is to be evaluated beyond the top- n ranks, one could also decide to measure prec@ $2n$, prec@ $3n$, or precision at some other point in the ranking, but the typical choice is to use the number of true outliers as cutoff value for measuring precision (CAMPOS *et al.*, 2016). A common alternative is the Receiver Operating Characteristic (ROC), which compares the candidate ranking against the binary ground truth by plotting the true positive rate against the false positive rate. Variants of these measures and more in-depth considerations about the external evaluation of unsupervised outlier detection results have been discussed, *e.g.*, by Schubert *et al.* (2012) and Campos *et al.* (2016).

For internal evaluation, which is the focus of the current chapter, the *top- n* IREOS index, preliminarily introduced in the author’s master’s work (MARQUES, 2015; MARQUES *et al.*, 2015), is, to our knowledge, the first internal validation index for unsupervised outlier detection. The absence of any index prior to IREOS had been noted as a gap in the literature that could hinder the development of advanced ensemble selection methods (ZIMEK; CAMPELLO; SANDER, 2013), but the potential applications of internal measures are far more diverse. Most fundamental is the practical application of outlier detection methods where users would benefit from unsupervised estimates of the quality of a solution provided by some method. After all, the availability of labeled data required by external evaluation measures is not consistent with the premises of unsupervised learning, and the commonly practiced external evaluation of unsupervised outlier detection algorithms makes sense only when comparing performances of algorithms in controlled experiments.

5.3 Internal Evaluation of Outlier Detection

5.3.1 Problem Statement

Let $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ be an unlabeled dataset containing N d -dimensional feature vectors, $\mathbf{x}_i$, and assume that one or more unsupervised outlier detection algorithms will produce, for this dataset, a collection Ω of n_Ω candidate outlier detection solutions, $\Omega = \{\omega_1, \dots, \omega_{n_\Omega}\}$, which one wants to evaluate in the absence of labels. Solutions $\omega_{(\cdot)}$ produced by unsupervised outlier detection algorithms can be given in different formats. The most common format is a scoring $\mathbf{y} = \{y_1, \dots, y_N\}$, $y_i \in \mathbb{R}^+$, where y_i represents the outlier score associated with the data object/observation $\mathbf{x}_i$, which reflects the degree of outlierness of $\mathbf{x}_i$. This type of solution allows objects $\mathbf{x}_i$ to be sorted and ranked according to their degree of outlierness y_i . When the number

of outliers n is known, one can establish a threshold in the ranking in order to select a subset $\mathbf{S} \subset \mathbf{X}$, $|\mathbf{S}| = n$, containing the top- n objects that are labeled as outliers. When represented in this format, we refer to $\omega_{(\cdot)}$ as binary, top- n solutions.

Given a collection Ω of candidate solutions ω_i , whether they are scoring solutions or top- n solutions, we want to independently and quantitatively measure the quality of each individual candidate solution, *e.g.*, in order (i) to assess their statistical significance when compared to the null hypothesis of a random solution; or (ii) to compare them in relative terms so that the best candidates, corresponding to more suitable models (algorithms, parameters), can be selected.

5.3.2 Preliminary Attempt: A Baseline Index

We start from the intuition that an outlier is an observation that is to some extent farther off and can therefore be more easily separated (discriminated) from other observations than an inlier. Initially, we will consider the top- n outlier detection problem, where a binary labeling of n out of N data objects as outliers, corresponding to a better (worse) unsupervised outlier detection solution $\mathbf{S}$, is expected to be more (less) in conformity with this intuition. The basic problem is then to quantify how easy or difficult it is to separate each object $\mathbf{x}_i \in \mathbf{S}$ from other objects. In a good solution $\mathbf{S}$, consisting mostly of genuine outliers correctly detected by some method, the average degree of separability is expected to be high, whereas in a poor solution containing many false positives this average degree of separability should be lower.

We propose to assess the separability of individual data objects using a classifier. We advocate the use of a maximum margin classifier (SCHÖLKOPF; SMOLA, 2001; ZHU; HASTIE, 2005; HASTIE; TIBSHIRANI; FRIEDMAN, 2013), as this type of classifier is able to quantify how distant each object is from the decision boundary while trying to maximize the margin of separability between this boundary and the instances of different classes. This idea is illustrated in Figure 24. Figures 24a, 24b, and 24c highlight different objects labeled as an outlier (red square) in different hypothetical outlier detection solutions. In Figure 24a, the highlighted object, a genuine global outlier, is far away from a maximum margin classification boundary (dashed line) that discriminates it from the other objects. In Figure 24b, the highlighted object is arguably a local outlier (w.r.t. the neighboring cluster) and the margin is narrower but still wider than that in Figure 24c. In the case of Figure 24c, the highlighted object is undoubtedly an inlier and not only the margin is very narrow but also the decision boundary needs to be nonlinear (*i.e.*, more complex).

The fact that the decision boundary needs to be nonlinear to separate certain objects (as in the example in Figure 24c) implies that a nonlinear maximum margin classifier is required for our purpose, such as Nonlinear SVMs or Kernel Logistic Regression (SCHÖLKOPF; SMOLA, 2001; ZHU; HASTIE, 2005; HASTIE; TIBSHIRANI; FRIEDMAN, 2013). These classifiers use a kernel function to transform the original (possibly non-linearly separable) problem into a

Figure 24 – Illustrative dataset: three different objects labeled as outliers.

linearly separable one. One of the most effective and popular kernel functions is the radial basis kernel, given by:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}. \quad (5.1)$$

The term γ , which is inversely proportional to the width of such a Gaussian-shaped kernel, is positively related to the flexibility (degree of nonlinearity) of the decision boundary of the corresponding classifiers. In other words, the discrimination capacity of a kernel-based classifier is positively dependent on γ . As a special case it approaches a linear classifier as γ approaches zero. The effect of γ is similar to that of the order of a polynomial kernel function, starting from linear for first order and getting more and more non-linear as the order increases.

In practical classification tasks, γ can be used to control the compromise between the performance of the classifier on the training data versus on test data. Here, however, we are not interested at all in the classifier itself or its performance on new, unseen data. We use a classifier merely to measure the degree of difficulty when trying to discriminate between one individual data object and the other data objects. The key observation to achieve this without having to specify a particular value for γ as a parameter is that our original premise tends to hold true, to a lesser or greater extent, no matter the value of γ . In other words, the fundamental assumption “*the more outlierish an object is, the easier is it to discriminate from others*” is expected to be observed for different values of γ , although the contrast between easier and more difficult cases may change. This is illustrated in Figure 25a. We vary the value of γ from zero up to a maximum value γ_{max} (for which all the objects labeled as outliers — Figures 24a, 24b, 24c — can be individually discriminated from all the others by using a kernel-based classifier). The values along the curves (vertical axis) stand for a measure $p(\mathbf{x}_j, \gamma)$ that quantifies in a normalized interval how far each object $\mathbf{x}_j$ is from the decision boundary. For all values of γ , the two outliers are distinctly farther away from the decision boundary than the inlier.

Thus, we do not need to choose a particular value of γ . Instead, we can measure the overall separability of an object $\mathbf{x}_j$ by computing the *area under the curve* (AUC) over the

Figure 25 – (25a) curves of separability for a maximum margin classifier for each of the objects labeled as outliers in Figure 24; (25b) weighted average separability curves of two top- n outlier detection solutions composed of the $n = 2$ outliers (local and global) of Figure 24, differing only in the outlier scores given to them in each of the solutions.

interval of γ values, *i.e.*,

$$\int_{\gamma=0}^{\gamma_{\max}} p(\mathbf{x}_j, \gamma). \quad (5.2)$$

Our final goal is, though, to evaluate the separability across the *collection* of data objects labeled as outliers in a given solution $\mathbf{S}$. We therefore take the average curve of separability for those objects in $\mathbf{S}$, *i.e.*,

$$\bar{p}(\gamma) = \frac{1}{n} \sum_{\mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma), \quad (5.3)$$

and then compute the area under this curve to get a single number, *i.e.*,

$$\int_{\gamma=0}^{\gamma_{\max}} \bar{p}(\gamma). \quad (5.4)$$

This value can be trivially normalized in $[0, 1]$ by dividing it by its maximum possible value, $\gamma_{\max}$, thus giving rise to a first, preliminary index,

$$I(\mathbf{S}) = \frac{1}{\gamma_{\max}} \int_{\gamma=0}^{\gamma_{\max}} \bar{p}(\gamma). \quad (5.5)$$

As in practice classifiers need to be trained to compute $\bar{p}(\gamma)$ for each γ , we discretize the interval $[0, \gamma_{\max}]$ into a finite number of values for γ , from $\gamma_1 = 0$ to $\gamma_{n_\gamma} = \gamma_{\max}$. A baseline index can thus be computed (within $[0, 1]$) as:

$$I(\mathbf{S}) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{1}{n} \sum_{\mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma_l) \right). \quad (5.6)$$

Non-binary top- n solutions

Typically, algorithms that produce top- n outlier solutions also provide a ranking of the objects according to their score of outlierness as estimated by the algorithm. Even though it is clear that the best solution must rank all outliers before all inliers, while the worst solution would rank all inliers before all outliers, there are $n!$ different best top- n solutions that would be equally well rated by the index in Equation (5.6). These solutions correspond to rank permutations of the same top- n outliers. The reason why our preliminary index in Equation (5.6) does not distinguish between them is that it does not use the information of the outlier scores or ranks of the top- n data objects, but rather only the data objects themselves (the subset $\mathbf{S}$). However, the optimal outlier solution should rank more obvious or clear outliers before less obvious outliers, before those borderline cases that could be outliers or inliers, and so on. This issue has been addressed (SCHUBERT *et al.*, 2012) in the context of the *external* evaluation of outlier detection, where the traditional external measures such as ROC AUC and $prec@n$ cannot make such a distinction either, even though they have access to a ground truth.

In order to address this problem in our context of *internal* or *relative* evaluation of outlier detection, we propose to introduce weights in Equation (5.3), which defines the average curve of separability. This way, Equation (5.3) becomes a weighted average curve of separability, where a weight w_j is given to the separability values $p(\mathbf{x}_j, \gamma)$ of each object $\mathbf{x}_j$, and thus different rankings or scorings of the same set of outliers can be distinguished:¹

$$\bar{p}(\gamma) = \frac{\sum_{j: \mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma) w_j}{\sum_{j: \mathbf{x}_j \in \mathbf{S}} w_j}, \quad (5.7)$$

The use of weights for *external* outlier evaluation has been discussed by Schubert *et al.* (2012), who advocate the usefulness of algorithms that provide outlier scores as a result, pointing out, however, that the direct comparison of such results from different algorithms is far from trivial because different algorithms usually produce scores in completely different scales. For this reason, some kind of normalization is unavoidable when comparing solutions provided by different algorithms. Following the recommendation of Schubert *et al.* (2012), we use the outlier scoring normalization framework proposed by Kriegel *et al.* (2011), as this framework comprises statistically sound methods based on distribution fitting that can accommodate a large variety of outlier detection approaches. By using this framework, the original outlier scores ($\mathbf{y}$) produced by a given algorithm can be transformed into the interval $[0, 1]$ in a way that they can be interpreted as outlier probabilities. The resulting normalized outlier scores, $w_{(\cdot)}$, can then be used as the weights associated to objects $\mathbf{x}_j$, as shown in Equation (5.7), irrespective of the algorithm that produced the solution under evaluation. The interpretation of the weights as probabilities will

¹ Notice that the use of a weighted average rather than a weighted sum prevents outlier solutions from artificially achieving higher weighted separability values by indiscriminately increasing the weights.

also be particularly useful when enhancing our basic notion of separability discussed above towards a more sophisticated model for outlier evaluation, to be discussed later, in section 5.3.3.

The intuition behind the use of weights in Equation (5.7) as described above is the following: when evaluating an ideal or optimal solution, both the outlier scores and the separabilities of the more obvious outliers are expected to be higher than those of the other objects, therefore their product will be higher. On the other hand, in a suboptimal solution, where the top- n objects may be the same as the optimal solution but those with highest separability are not the ones with the highest scores (weights) and vice-versa, the product will be penalized. In summary, we expect that solutions assigning higher (lower) outlier scores to objects with higher (lower) separabilities will result in larger AUC values as computed by the weighted version of Equation (5.6), given by:

$$I(\mathbf{S}, \mathbf{y}) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{\sum_{j: \mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma) w_j}{\sum_{j: \mathbf{x}_j \in \mathbf{S}} w_j} \right). \quad (5.8)$$

Notice that the index in Equation (5.8), in addition to the subset of top- n objects, $\mathbf{S}$, also requires as argument the outlier scores, $\mathbf{y}$, which are used to compute the weights $w_{(\cdot)}$ following the method by [Kriegel et al. \(2011\)](#). Hereafter, we refer to this type of solution as *non-binary top- n solutions*.

An illustrative example corresponding to two hypothetical non-binary top- n outlier detection solutions is provided in Figure 25b. Both hypothetical solutions are composed of the $n = 2$ outliers (global and local) highlighted in Figure 24, differing only in the outlier scores assigned to them in each of the solutions. The optimal solution ranks the global outlier (with an outlier score of 1) before the local outlier (with an outlier score of 0.75), as opposed to the suboptimal solution that ranks the local outlier first (with an outlier score of 1) and the global outlier next (with an outlier score of 0.75). These two different solutions would be equally well rated by the original baseline index in Equation (5.6). However, using the proposed weighted average curve of separability, the index can properly distinguish these slightly different solutions, as it can be seen from the corresponding weighted average curves of separability in Figure 25b. As expected, the optimal solution has a larger AUC than the suboptimal solution, which corresponds to a larger value of the new baseline index, $I(\mathbf{S}, \mathbf{y})$, in Equation (5.8).

Evaluation of full outlier scorings

In many practical applications of unsupervised outlier detection, the number of outliers in a dataset, n , is unknown, and all that is provided by an algorithm is a scoring $\mathbf{y} = \{y_1, \dots, y_N\}$, $y_i \in \mathbb{R}^+$, containing the outlier scores y_i associated with every data object $\mathbf{x}_i \in \mathbf{X}$ in the dataset. An interesting aspect of our baseline index in Equation (5.8) is that it can be straightforwardly generalized to this *full scoring* evaluation scenario, by just setting $n = N$, which implies $\mathbf{S} = \mathbf{X}$.

From this observation, we can write our baseline index in a more general form that encompasses the top- n scenarios previously discussed as particular cases. Specifically, given an outlier detection solution to be evaluated, ω , our general baseline index is as follows:

$$I(\omega) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{\sum_{j=1}^N p(\mathbf{x}_j, \gamma) w_j}{\sum_{j=1}^N w_j} \right). \quad (5.9)$$

In the general case of *full scoring* evaluation, the outlier solution ω corresponds to a scoring $\mathbf{y} = \{y_1, \dots, y_N\}$ for all objects in the dataset, the corresponding probability weights $w_{(\cdot)}$ can be computed from these scores, and these weights can be directly applied into Equation (5.9). It is worth noticing that, in this scenario, the use of weights $w_{(\cdot)}$ normalized as probabilities following the approach by [Kriegel et al. \(2011\)](#) is particularly useful because the majority of the data objects, as clear inliers, will tend to have a weight of zero, thus having no undesired (over) influence on the results (which could otherwise occur if the raw outlier scores $y_{(\cdot)}$ were used instead, as these typically have a very long tail of spurious small values for inliers, which, when aggregated, could dominate the results).

In the case previously referred to as *non-binary top- n solutions*, in addition to a scoring $\mathbf{y}$ the outlier solution ω also contains the subset $\mathbf{S}$ of top- n outliers, *i.e.*, $\omega = (\mathbf{S}, \mathbf{y})$. In this case, the weights $w_{(\cdot)}$ corresponding to those objects outside $\mathbf{S}$ (*i.e.*, not in the top- n) must be set to zero before the index is computed with Equation (5.9). By enforcing $w_j = 0 \ \forall j : \mathbf{x}_j \notin \mathbf{S}$, Equation (5.9) becomes equivalent to Equation (5.8). In the even simpler case of a (binary) top- n solution that does not consider scores associated to the top- n outliers, *i.e.*, $\omega = \mathbf{S}$, if, in addition to setting to zero the weights of objects labeled as inliers, the weights of the top- n outliers are set to 1 ($w_j = 1 \ \forall j : \mathbf{x}_j \in \mathbf{S}$), then Equation (5.9) reduces to Equation (5.6).

5.3.3 IREOS Index

Intuitions missing in the baseline index

Our preliminary baseline index introduced in section 5.3.2 may work satisfactorily in various application scenarios. Conceptually, however, it does not capture two basic intuitions that we judge important in the realm of outlier detection. Both are related to the possible presence of *clumps* of data objects in the dataset. Clumps, or particles, are subsets of objects lying in the same region of the data space, relatively closer to each other than they are from other objects, but too small to be deemed a cluster. They may exist for different reasons, mainly: (i) just by chance, *e.g.*, in datasets with background noise following a Poisson process; or (ii) as a result of anomalies whose instances are relatively rare but tend to be similar to each other, *e.g.*, some genetic mutations or certain types of frauds. Although the semantics behind the possible

interpretation of such clumps as outliers would be different, namely, noise in the first case and micro-clusters in the second, in both cases the analyst may not want to miss the corresponding objects as potential outliers for further investigations.

Unfortunately, what is a clump can depend on both the application domain and a user's expectation. If we do not want our measure to be tied to specific evaluation perspective that prescribes a clump size, we thus need a mechanism for a user to express in different application scenarios what they judge to be a set of similar objects that is "too small" to be interpreted as a cluster. Therefore, in contrast to the common practice of avoiding parameters in evaluation indexes for unsupervised learning, we advocate here that for outlier detection it is nonetheless important to provide the users with an optional control mechanism to adjust their expectations about clump sizes. Given a certain expectation about what a maximum clump size should be, we should model an evaluation index so that it is able to differentiate between objects inside clusters and objects in isolated clumps. This is the first intuition that is missing in our baseline index as it was defined in section 5.3.2. In order to capture this intuition, we define a maximum clump size, m_{cl} , as an optional control parameter for exploratory data analysis, to be incorporated in our index.

The second intuition that is not captured by the preliminary index is also related to clumps. While it is clear that the evaluation of each object labeled as an outlier and, accordingly, the whole index, should be negatively affected by the presence of other objects nearby (*e.g.*, in a clump), it is intuitive that such a negative impact should be more severe if the nearby objects are assigned a different label (*i.e.*, they are actually deemed inliers). Consider the example in Figure 26, which corresponds essentially to the same dataset as Figure 24 except for an additional object placed near the global outlier at the left bottom corner. On the left, this object's label (inlier) appears to be inconsistent with the label of the original object close by (outlier), and the index should be negatively affected. On the right, even though the original object is less of an outlier now in the presence of the additional object, their common labeling as outliers is more consistent as both objects can be seen as a clump, so the negative impact of the presence of the new object should be smaller.

Incorporating the missing intuitions

In order to capture both desired intuitions, we propose the use of classifiers with *soft margins* to compute the values of separability $p(\mathbf{x}_j, \gamma)$ required by our index. The use of soft margin models is common practice both in the literature as well as in real-world applications when using maximum margin classifiers, such as SVMs and Kernel Logistic Regression (SCHÖLKOPF; SMOLA, 2001; ZHU; HASTIE, 2005; HASTIE; TIBSHIRANI; FRIEDMAN, 2013). By making use of a soft margin, these classifiers allow the misclassification of objects at the price of a penalty term P_i that is incorporated into the original objective of margin maximization. Such a

Figure 26 – Dataset of Figure 24 with an additional object at the lower left corner: (26a) labeled inlier; (26b) labeled outlier.

term is typically in the form:

$$P_t = C \sum_{j=1}^N \xi(\mathbf{x}_j), \quad (5.10)$$

where C is a constant and $\xi(\mathbf{x}_j)$ stands for the individual penalty component associated with object $\mathbf{x}_j$. The farther an object $\mathbf{x}_j$ is from the margin boundary on the wrong side, the greater the value of $\xi(\mathbf{x}_j)$.² The constant C controls the overall cost of penalties. In classification problems, this is a key parameter used to adjust the compromise between under- and overfitting. Here, since we are not interested in the performance of the classifiers for new data, this constant is not critical and should only be big enough so that each object in the dataset can be discriminated from others by these classifiers when following the procedure to compute the proposed index.

Soft margin classifiers allow the use of a generalized penalty component that can assign different costs $C(\mathbf{x}_j)$ to different objects, rather than a single, uniform cost C :

$$P_t = \sum_{j=1}^N C(\mathbf{x}_j) \xi(\mathbf{x}_j). \quad (5.11)$$

A suitable design of terms $C(\mathbf{x}_j)$ makes it possible to simultaneously incorporate both desired intuitions that are missing in our baseline index, related to the modeling of clumps. For the sake of simplicity, let's initially consider the simplest case involving *binary top- n solutions* $\mathbf{S}$, where n out of N objects have been labeled as outliers, and the others have been labeled as inliers. In order to capture the differences between the two distinct scenarios in Figure 26, we can assign full margin violation cost C to the objects labeled as inliers yet only a fraction $\beta \in [0, 1]$

² For SVMs, $\xi(\mathbf{x}_j)$ is zero when $\mathbf{x}_j$ lies on the correct side of the margin boundary. For Kernel Logistic Regression, all objects (rather than only support vectors) can influence the decision boundary and $\xi(\mathbf{x}_j)$ can be non-null but tending to zero as $\mathbf{x}_j$ moves away from the margin boundary on the correct side.

of C to the objects labeled as outliers, *i.e.*, $C(\mathbf{x}_j) = C$ or $C(\mathbf{x}_j) = \beta \cdot C$, depending on whether $\mathbf{x}_j$ has been labeled as an inlier or an outlier in the solution $\mathbf{S}$ under evaluation, respectively. For $\beta = 1$, the method reduces to the ordinary case where objects are treated equally no matter their labels, and thus the separability of the leftmost outlier in Figures 26a and 26b would be indistinguishable. In the other extreme, $\beta = 0$, objects labeled as outliers can be misclassified for free. Notice that, when evaluating the separability of a specific object, this is equivalent to removing all other objects labeled as outliers from the dataset. As a consequence, the separability of the leftmost outlier would be much larger in Figure 26b (where the neighboring outlier can be ignored by the margin) than in Figure 26a (where the neighboring inlier, which is subject to full penalty, forces a much tighter margin). The choice of β would therefore tune the influence of other objects depending on their assigned labels and, as such, it addresses our *second desired intuition*.

In order to capture our *first intuition*, the modeling of possible clumps by defining a maximum clump size, m_{cl} , we can set the fraction of the penalty cost C as $\beta = 1/m_{cl}$. This way, we are left with m_{cl} as a single, optional control parameter in our evaluation method. It is optional because, by setting $m_{cl} = 1$, the method reduces to the particular case where clumps are not modeled and the same, full penalty cost is assigned to all objects. As m_{cl} increases, the objects in a clump will individually affect less and less each other's measure of separability, and a larger number of nearby objects will be needed to get a certain negative impact. Notice that, in the top- n problem, by setting $\beta = 1/m_{cl}$ one needs m_{cl} objects labeled as outliers to have the same effect as a single inlier. Also, notice that it would be contradictory to set $m_{cl} > n$, as no more than n objects can be labeled as outliers in a top- n problem. By considering this conceptual upper bound, one gets $1 \leq m_{cl} \leq n$. Except when $m_{cl} = 1$, the separability of each object depends on the labels of the other objects and, therefore, seeking a solution that maximizes the proposed index (rather than using it to assess a given solution) would hardly be computationally feasible: in principle, for the top- n problem, it would demand an exhaustive search in a space of size $\binom{N}{n}$, where typically $n \ll N$. In the more general problem of *full outlier scoring evaluation*, discussed next, the size of the space would expand even further due to the infinite number of different possible assignments of outlier scores to each object (rather than a binary inlier/outlier labeling).

We can extend the above formulation for modeling clumps to the more general evaluation problem involving *full outlier scoring solutions*. To achieve this while keeping the desired properties of the model previously discussed for the top- n case, we can incorporate the normalized outlier scores (weights) $w_{(\cdot)}$ into the cost components, as $C(\mathbf{x}_j) = \beta^{w_j} \cdot C$. For an object $\mathbf{x}_j$ with maximum probability of being an outlier according to the solution under evaluation, *i.e.* $w_j = 1$, it follows that $C(\mathbf{x}_j) = \beta \cdot C$. In the other extreme, for an object with maximum probability of being an inlier, *i.e.*, $w_j = 0$, we have $C(\mathbf{x}_j) = C$. For intermediate values, $0 < w_j < 1$, the higher the outlier score, the higher the penalty cost $C(\mathbf{x}_j)$. This way, we keep the original intuition in this extended, full scoring evaluation model, while having the binary, top- n model clearly as a particular case.

In the general case, therefore, the optional model for clumps to be incorporated into our baseline index, through the use of penalty terms associated with soft margin classifiers as in Equation (5.11), is therefore:

$$C(\mathbf{x}_j) = \left(\frac{1}{m_{cl}} \right)^{w_j} \cdot C, \quad (5.12)$$

where m_{cl} is the maximum clump size, as a user-defined parameter, and C is a large constant.

Summary and algorithm

In brief, IREOS is summarized as follows: like the baseline index (section 5.3.2), IREOS is computed using Equation (5.9). However, we make use of classifiers with soft margins in order to compute the terms $p(\mathbf{x}_j, \gamma)$ in that equation, with the cost for margin violations set up as in Equation (5.12). A high level pseudo code for computing IREOS for a set Ω of multiple outlier detection solutions (as, *e.g.*, for model selection) is given in Algorithm 1.

As for the classifier to be used in practice, our method is not tied to any specific soft margin classifier. Kernel Logistic Regression (KLR) (ZHU; HASTIE, 2005; HASTIE; TIBSHIRANI; FRIEDMAN, 2013), which we have used for all the experiments reported in this chapter, offers the following benefits: (i) it automatically provides $p(\mathbf{x}_j, \gamma)$ as the probability that object $\mathbf{x}_j$ belongs to the positive (outlier) class; (ii) these terms are not only provided directly as a byproduct of the classifier, but they are naturally normalized (as probabilities) within $[0, 1]$; and (iii) KLR is a classifier known to be robust even in the presence of imbalanced classes and small amounts of training data.

5.3.4 Adjustment for Chance

The IREOS index as described above is ready to be used in practice if one is only interested in comparing in *relative terms* a set of different candidate solutions, *e.g.* for model selection. However, the interpretation of the index for individual solutions, *e.g.* for statistical validation, can be misleading. The reason is that IREOS will provide a certain positive value even when evaluating purely random solutions. The situation is worsened by the fact that such a value is data dependent. In fact, note from Figure 25a that even inliers will exhibit a non null value for the AUC of separability. This prevents interpreting and assessing the statistical relevance of a given result in *absolute terms*, which requires the index to be adjusted for chance. Here, we follow the classic statistical framework for chance adjustment, *i.e.*,

$$I_{adj}(\omega) = \frac{I(\omega) - E\{I\}}{I_{max} - E\{I\}}, \quad (5.13)$$

where $I_{adj}(\omega)$ is the resulting (adjusted) index, $I(\omega)$ is the original index in Equation (5.9), $I_{max} = 1$ is the maximum value that the index can take, and $E\{I\}$ is its expected value assuming that the outlier weights are randomly assigned to the objects (for scoring solutions), or assuming

Algorithm 1 – IREOS

```

Procedure IREOS( $\mathbf{X}, \Omega, m_{cl}$ )
  foreach ( $\omega \in \Omega$ ) do
    switch  $\omega$  do
      case is binary top-n solution do                                     //  $\omega = \mathbf{S}$ 
        if  $\mathbf{x}_j \in \mathbf{S}$  then
          |  $\mathbf{w}[\mathbf{x}_j] = 1$ 
        else
          |  $\mathbf{w}[\mathbf{x}_j] = 0$ 
        end
      end
      case is non-binary top-n solution do                               //  $\omega = (\mathbf{S}, \mathbf{y})$ 
         $\mathbf{w} = \text{NormalizeScores}(\mathbf{y})$ 
        if  $\mathbf{x}_j \notin \mathbf{S}$  then
          |  $\mathbf{w}[\mathbf{x}_j] = 0$ 
        end
      end
      case is scoring solution do                                       //  $\omega = \mathbf{y}$ 
        |  $\mathbf{w} = \text{NormalizeScores}(\mathbf{y})$ 
      end
    end

    /*  $\gamma_{max}$  = value of  $\gamma$  required by the classifier to separate from the other
       objects every object labeled as an outlier in all solutions  $\omega \in \Omega$ ; for scoring
       solutions, objects are considered outliers if their weight  $w_j$  is greater than
       0.5 (i.e., outlier probability > 50%) */
     $\gamma_{max} = \gamma \mid \text{Classifier}(\mathbf{X}, \mathbf{x}_j, \mathbf{w}, m_{cl}, \gamma) > 0.5, \forall \mathbf{x}_j \mid w_j > 0.5, \forall \omega \in \Omega$ 
     $\text{setOfGammas} = [0, \gamma_{max}]$  discretized into  $n_\gamma$  values
    foreach ( $\gamma \in \text{setOfGammas}$ ) do
      foreach ( $\mathbf{x}_j \in \mathbf{X}$ ) do
        |  $\text{prob}[\mathbf{x}_j] = \text{Classifier}(\mathbf{X}, \mathbf{x}_j, \mathbf{w}, m_{cl}, \gamma)$            // Separability  $p(\mathbf{x}_j, \gamma)$ 
      end
       $\text{avgProb}[\gamma] = \text{WeightedAverage}(\text{prob}, \mathbf{w})$                    // Inner part of Equation (5.9)
    end
     $\text{ireos}[\omega] = \text{NormAUC}(\text{avgProb}, \text{setOfGammas})$                  // Outer part of Equation (5.9)
  end
end

```

that the n data objects labeled as outliers in a top- n solution are chosen randomly. For random solutions, I_{adj} is expected to take values around zero. The maximum is still 1, but the index now can take negative values to indicate solutions even worse than what one would expect to obtain by chance.

Exact computation — scoring solutions

Initially, we consider the evaluation of *full scorings*, where a random solution according to the adopted null model assumes that the outlier weights $w_{(\cdot)}$ are randomly and independently assigned to objects following a uniform distribution within $[0, 1]$. In order to adjust IREOS for chance, all we need is to derive term $E\{I\}$ in Equation (5.13) for this null model. Notice that this

term can be written from Equation (5.9) as:

$$E\{I\} = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} E\{\bar{p}(\gamma_l)\}, \quad (5.14)$$

where

$$\bar{p}(\gamma) = \frac{\sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot w_j}{\sum_{j=1}^N w_j} = \sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot \tilde{w}_j, \quad (5.15)$$

$$\tilde{w}_j = \frac{w_j}{\sum_{j=1}^N w_j}. \quad (5.16)$$

Term $E\{\bar{p}(\gamma)\}$ in Equation (5.14) can be written from Equation (5.15) as:

$$E\{\bar{p}(\gamma)\} = E\left\{\sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot \tilde{w}_j\right\}. \quad (5.17)$$

The random scoring solutions $w_{(\cdot)}$ are independent from the data objects, but the objects' separabilities $p(\cdot, \cdot)$ will depend on $w_{(\cdot)}$ when the optional modeling of clumps is in place, *i.e.*, when $m_{cl} > 1$ in Equation (5.12). In this case, it is hard to derive $E\left\{\sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot \tilde{w}_j\right\}$ analytically, due to the complex relation between $p(\cdot, \cdot)$ and $w_{(\cdot)}$, which depends on the kernel-based soft margin classifier formulation and optimization solution. For this reason, we initially assume the scenario where the optional model of clumps is not in place, *i.e.*, $m_{cl} = 1$.

Recall that, when $m_{cl} = 1$, the classifiers just try to discriminate between each data object $\mathbf{x}_j$ and the other objects, no matter their outlier labeling or scorings. In this case, $p(\cdot, \cdot)$ does not depend on $w_{(\cdot)}$ and, thus, Equation (5.17) can be written as:³

$$E\{\bar{p}(\gamma)\} = \sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot E\{\tilde{w}_j\}. \quad (5.18)$$

Term $E\{\tilde{w}_j\}$ can be computed analytically as follows. From Equation (5.16), it is clear that $\sum_{j=1}^N \tilde{w}_j = 1$, thereby:

$$E\left\{\sum_{j=1}^N \tilde{w}_j\right\} = \sum_{j=1}^N E\{\tilde{w}_j\} = E\{1\} = 1, \quad (5.19)$$

and since the weights w_j are i.i.d.:

$$E\{\tilde{w}_1\} = E\{\tilde{w}_2\} = \dots = E\{\tilde{w}_N\}, \quad (5.20)$$

which in Equation (5.19) yields:

$$N \cdot E\{\tilde{w}_j\} = 1, \quad (5.21)$$

³ When $m_{cl} = 1$, the separabilities can be pre-computed for each data object, irrespective of the outlier solution under evaluation and, accordingly, they become constant terms in the adopted null model.

and, accordingly:

$$E\{\tilde{w}_j\} = \frac{1}{N}. \quad (5.22)$$

Substituting (5.22) into Equation (5.18), we have the following result:

$$E\{\bar{p}(\gamma)\} = \sum_{j=1}^N p(\mathbf{x}_j, \gamma) \cdot E\{\tilde{w}_j\} = \frac{1}{N} \sum_{j=1}^N p(\mathbf{x}_j, \gamma), \quad (5.23)$$

which in Equation (5.14) yields:

$$E\{I\} = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{1}{N} \sum_{j=1}^N p(\mathbf{x}_j, \gamma_l) \right). \quad (5.24)$$

Exact computation — top- n solutions

For binary top- n solutions, the previous null model is no longer applicable as there are not any scores available. The adopted null model in this case assumes that a random solution consists of a subset of n objects to be labeled as outliers, which are drawn from the data set $\mathbf{X}$ with uniform probability. We can therefore interpret such a random solution as a sample $\mathbf{S}_R \subset \mathbf{X}$ of size $|\mathbf{S}_R| = n$ taken without replacement from a population $\mathbf{X}$ with size $|\mathbf{X}| = N$.

Recall that, in binary top- n solutions, the average separability term $\bar{p}(\gamma)$ is not weighted, and it is computed over the separabilities of the top- n objects only, *i.e.*:

$$\bar{p}(\gamma) = \frac{1}{n} \sum_{\mathbf{x}_j \in \mathbf{S}_R} p(\mathbf{x}_j, \gamma), \quad (5.25)$$

from which we can take the expectation as:

$$E\{\bar{p}(\gamma)\} = \frac{1}{n} \sum_{\mathbf{x}_j \in \mathbf{S}_R} E\{p(\mathbf{x}_j, \gamma)\} = E\{p(\mathbf{x}_j, \gamma)\}. \quad (5.26)$$

This is an instance of the well-known result that the expected value for the mean of an i.i.d. sample of size n is the mean of the population. Here, for a given γ , our (finite) population consists of the N precomputed values $p(\mathbf{x}_j, \gamma)$ for all data objects $\mathbf{x}_j$ in the database $\mathbf{X}$. In fact, for $m_{cl} = 1$, the separabilities $p(\cdot, \cdot)$ depend only on the data, not on any particular realization $\mathbf{S}_R$ of random candidate outliers, and therefore they can be independently precomputed. Taking their average gives the exact value for $E\{\bar{p}(\gamma)\}$, *i.e.*:

$$E\{\bar{p}(\gamma)\} = E\{p(\mathbf{x}_j, \gamma)\} = \frac{1}{N} \sum_{j=1}^N p(\mathbf{x}_j, \gamma). \quad (5.27)$$

Substituting this equation into Equation (5.14) yields once again the very same result in Equation (5.24). In other words, even though the null models are different, the expected value of the index is the same, given by Equation (5.24), for both scoring or binary top- n solutions.

Statistical validation — top- n solutions

The variance is not needed for the adjustment for chance in Equation (5.13), but it can be useful for statistical validation when this type of validation is required. Since our index is given by a sum of random variables $\frac{1}{n_\gamma} \bar{p}(\gamma_i)$ over γ_i , we can compute the variance of the index as:

$$\text{Var}\{I\} = \text{Var}\left\{\frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \bar{p}(\gamma_l)\right\} = \frac{1}{n_\gamma^2} \sum_{l_1, l_2=1}^{n_\gamma} \text{Cov}(\bar{p}(\gamma_{l_1}), \bar{p}(\gamma_{l_2})), \quad (5.28)$$

which can also be rewritten equivalently as:

$$\text{Var}\{I\} = \frac{1}{n_\gamma^2} \sum_{l=1}^{n_\gamma} \text{Var}\{\bar{p}(\gamma_l)\} + \frac{2}{n_\gamma^2} \sum_{l_1=1}^{l_2-1} \sum_{l_2=2}^{n_\gamma} \text{Cov}(\bar{p}(\gamma_{l_1}), \bar{p}(\gamma_{l_2})). \quad (5.29)$$

This equivalent form emphasizes the possible lack of independence of $\bar{p}(\gamma_i)$ over γ_i , specifically when the second term is not null. For the first term, it follows from Equation (5.25) that:⁴

$$\text{Var}\{\bar{p}(\gamma)\} = \frac{1}{n^2} \sum_{\mathbf{x}_j \in \mathbf{S}_R} \text{Var}\{p(\mathbf{x}_j, \gamma)\} = \frac{1}{n} \text{Var}\{p(\mathbf{x}_j, \gamma)\}, \quad (5.30)$$

i.e., the variance of the sample mean is the variance of the population over the sample size. Analogously, for the covariance one has

$$\text{Cov}(\bar{p}(\gamma_1), \bar{p}(\gamma_2)) = \frac{1}{n} \text{Cov}(p(\mathbf{x}_j, \gamma_1), p(\mathbf{x}_j, \gamma_2)), \quad (5.31)$$

Equations (5.30) and (5.31), which are required by Equation (5.29), can both be exactly computed once we have precomputed the whole population $p(\cdot, \cdot)$, which is possible when $m_{cl} = 1$. In this case, provided that the sample size is not critically small, the Central Limit Theorem (CLT) ensures that, for each γ_i , the sample mean $\bar{p}(\gamma_i)$ follows at least approximately a Normal distribution, *i.e.*,

$$\bar{p}(\gamma) \sim \mathcal{N}(E\{\bar{p}(\gamma)\}, \text{Var}\{\bar{p}(\gamma)\}). \quad (5.32)$$

This means that, for random binary solutions $\mathbf{S}_R$ and $m_{cl} = 1$, our index in Equation (5.6) is given by a sum of normally distributed variables $\frac{1}{n_\gamma} \bar{p}(\gamma_i)$ over γ_i . The sum of normally distributed random variables

$$\mathbf{X} \sim \mathcal{N}(\mu_{\mathbf{X}}, \sigma_{\mathbf{X}}^2) \quad (5.33)$$

and

$$\mathbf{Y} \sim \mathcal{N}(\mu_{\mathbf{Y}}, \sigma_{\mathbf{Y}}^2) \quad (5.34)$$

⁴ Since the population is of a finite size (N), though, when considering sampling without replacement and sample sizes n significantly large w.r.t. N (*e.g.*, more than 5%), a finite population correction factor $(N - n)/(N - 1)$ can be used to adjust the computed variance (TRIOLA, 2007).

is also normally distributed, *i.e.*,

$$\mathbf{X} + \mathbf{Y} \sim \mathcal{N}(\mu_{\mathbf{X}} + \mu_{\mathbf{Y}}, \sigma_{\mathbf{X}+\mathbf{Y}}^2) \quad (5.35)$$

where (ROSS, 2009)

$$\sigma_{\mathbf{X}+\mathbf{Y}}^2 = \sigma_{\mathbf{X}}^2 + \sigma_{\mathbf{Y}}^2 + 2\text{Cov}(\sigma_{\mathbf{X}}, \sigma_{\mathbf{Y}}) \quad (5.36)$$

This leads to the interesting result that, in this particular setting (binary top- n solutions and $m_{cl} = 1$), our index IREOS, as a sum of sample means, will follow at least approximately a Normal distribution according to the CLT, *i.e.*, $I \sim \mathcal{N}(E\{I\}, \text{Var}\{I\})$, with mean $E\{I\}$ and variance $\text{Var}\{I\}$ computed in an exact way as described above.

Since we know such a mean and variance for the population, we thus can go beyond the ordinary adjustment for chance (Equation (5.13)) and perform statistical validation as well. Particularly, if we are given a certain outlier detection solution, $\mathbf{S}$, and the corresponding value for our adjusted index, $I_{adj}(\mathbf{S})$, we can assess the statistical significance of $I_{adj}(\mathbf{S})$ by means of a z -test. In this case, a p -value can be trivially computed based on the Normal assumption by contrasting $I_{adj}(\mathbf{S})$ against the null hypothesis of a random solution (TRIOLA, 2007).

Approximate computation via Monte Carlo

The exact computations described above presume $m_{cl} = 1$. For different evaluation setups, $p(\mathbf{x}_j, \gamma_i)$ can no longer be independently precomputed for each object $\mathbf{x}_j \in \mathbf{X}$, as the separability of a given object as assessed by the classifiers now depends also on the binary labels or the degrees of outlierness assigned to the other objects of the dataset, for each possible random solution. This means that, for a given γ_i , the size of our finite population expands from N to at least $\binom{N}{n}$ (for the binary top- n problem) and, as such, it can easily become intractable for exhaustive computations. Even when $m_{cl} = 1$, precomputing N terms $p(\mathbf{x}_j, \gamma_i)$ for each γ_i (*i.e.*, $N \cdot n_\gamma$ in total) may be computationally prohibitive for large databases as well, as each term demands to train an independent classifier.

To make adjustment for chance feasible when $m_{cl} > 1$ or N is large, and also to make statistical validation possible in evaluation scenarios other than the binary top- n case, we can use Monte Carlo simulations in order to estimate the relevant statistics rather than trying to compute them in an exact and exhaustive way. The idea is to sample a number n_{MC} of random outlier detection solutions whereby the desired statistical moments can be estimated. In particular, the expected value in Equation (5.13) can be directly estimated from the sample.

When statistical validation is needed, we also need to estimate the baseline distribution under the null hypothesis. There are different alternatives. For binary top- n solutions, if the normality assumption is evoked from the CLT, a parametric approach is possible based on a t -student distribution with the sample estimates for the mean and variance (*i.e.*, a t -test, which is known to be robust even when normality is not fully satisfied (BONEAU, 1960)). Alternatively,

p -values can be directly derived from observed histograms in a non-parametric way (JAIN; DUBES, 1988) for each candidate outlier solution ω_i .

The sample size, n_{MC} , clearly represents a trade-off between computational burden and accuracy. Larger (smaller) samples lead to more (less) accurate estimations yet from a larger (smaller) number of trained classifiers. Rather than setting the value for n_{MC} arbitrarily, one can also determine n_{MC} automatically, by specifying (i) a certain significance level as the probability that the sample mean will fall within, and (ii) a prespecified confidence interval around the population mean (TRIOLA, 2007).

5.3.5 Complexity

The asymptotic computational complexity of the algorithm depends on the complexity of the classifier, $\mathcal{O}(f(N, d))$, as a function of the database size N and dimensionality d . For each candidate solution ω , we need to compute IREOS (Equation (5.9)), which demands training $N \cdot n_\gamma$ classifiers in the case of *full scoring evaluation*, thus resulting in an overall complexity of $\mathcal{O}(N \cdot n_\gamma \cdot f(N, d))$ in this case. When the index is adjusted for chance, we may need to evaluate n_{MC} different random solutions in Monte Carlo simulations in order to estimate the expected index, leading to a complexity of $\mathcal{O}(n_{MC} \cdot N \cdot n_\gamma \cdot f(N, d))$. This is the complexity in the most general case. For *top- n evaluations*, we only need to train $n \cdot n_\gamma$ classifiers, where the usual assumption is that $n \ll N$, which reduces the complexity to $\mathcal{O}(n_{MC} \cdot n \cdot n_\gamma \cdot f(N, d))$. If Monte-Carlo simulations are not required (because $m_{cl} = 1$ or because statistical validation is not needed, as in the case of model selection, where all we need is to relatively compare different candidate solutions), the complexity is further reduced to $\mathcal{O}(n \cdot n_\gamma \cdot f(N, d))$.

Final remarks

Before demonstrating that the index developed in this chapter works well in practice, we want to point out that we are aware of a vulnerability that could potentially allow “adversarial” outlier detection methods to fool the index to some extent. Recall that the basic definition of the index for a full scoring (Equations 5.8 and 5.9) is a weighted average, where the average of separability values, weighted by an outlier detection method’s scores, is taken over the sum of the scores assigned by the method. It is easy to see that a method could get the maximum possible IREOS score, if it would somehow be able to detect the strongest outlier (in terms of separability), and would then assign this outlier a weight of 1 and all other data objects a weight of 0. While this property is not desirable, it is practically not very relevant, since outlier detection methods are typically not designed to fool an evaluation index. An alternative evaluation index that does not have this property can be based on a weighted Pearson correlation coefficient (with separability values as weights). However, the drawback of weighted correlation is that not all of the speed-up strategies developed in this section easily apply, making it computationally much

more demanding. We have implemented and experimented with an evaluation index based on weighted Pearson Correlation, and its performance in terms of quality is indeed comparable to the index proposed in this chapter, but not systematically better, and much worse in terms of runtime. Therefore, we recommend to use the index as proposed in this section, and we evaluate its performance in detail in the next section.

5.4 Evaluation

5.4.1 Datasets

We combine different strategies to annotate datasets used for evaluation, following statistical considerations, following the semantical notion of unusual classes, or following common procedures and examples from the literature.

Synthetic datasets

For experiments on synthetic data, we adopt high-dimensional benchmarking datasets previously introduced in the chapter 3. For the experiments performed in this chapter, points with Mahalanobis distances on the right tail of χ_d^2 distribution are seen as outliers, the farther on right, the more outlier.

Real world datasets and preliminary benchmarking experiments

The first collection of real world datasets is composed of 23 real datasets from a publicly available outlier detection repository that contains approximately 1000 variants of 23 main datasets (CAMPOS *et al.*, 2016). As these datasets are originally from clustering and classification tasks and, therefore, there is no guarantee that the ground truth labels of objects as inliers and outliers in these datasets are in conformity with the spatial distribution of the data (*i.e.*, in principle it is unknown if the semantic represented by the labels is properly captured in the feature space where the objects have been described), Campos *et al.* (2016) proposed two indexes to characterize the suitability of each candidate dataset for outlier detection benchmarking. One of the indexes is called *difficulty*, which measures the disagreement between the results of a collection of representative outlier detection methods and the ground truth labels, such that a high value of the index indicates that the methods have difficulty in detecting the objects labeled as outliers in the ground truth. Therefore, the ground truth labels of a dataset with a high difficulty value are not consistent with the notion of outliers as captured by a variety of algorithms, which may suggest that the labels are not aligned with the spatial distribution of the data. The second index proposed by Campos *et al.* (2016) is called *diversity*, which measures the disagreement

of a given collection of outlier methods with respect to the outlier scores that they assign to data objects labeled as outliers in the ground truth. A low diversity score combined with a low difficulty score suggests that the outliers in the ground truth can be detected by most algorithms (an easy dataset). A high diversity score combined with a high difficulty score can occur, *e.g.*, if most of the algorithms are unable to detect most of the outliers, but a few outliers, which vary from algorithm to algorithm, can be detected by each algorithm.

A limitation of the measures of difficulty and diversity mentioned above is that they rely on a collection of representative outlier detection algorithms. Therefore, if these measures suggest that a dataset is very hard, we do not know for sure if there is a problem with the existing algorithms only (in which case the dataset is suitable for benchmarking in studies involving future algorithms) or if there is a problem with the dataset (the labels are fully or partially inconsistent with the data distribution and, therefore, the dataset may not be suitable for benchmarking). For instance, a low diversity score combined with a high difficulty score could occur, *e.g.*, if the outliers in the ground truth are assigned low outlier scores (as inliers) by most algorithms, either because the algorithms are not good enough or because the objects labeled as outliers cannot be detected as such (or both). In order to help disambiguate these scenarios, we propose the use of IREOS as an additional, complementary measure of *hardness* for a dataset, which has the advantage of being algorithm independent, *i.e.*, unlike the measures of difficulty and diversity, it depends only on the data. To that end, we take the ground truth labels of a dataset as a candidate (binary, top- n) solution and compute IREOS, adjusted for chance, for this “solution”. Higher values (close to 1) would indicate that the ground truth labels are highly correlated with the separability of the objects labeled as outliers, while low values (around zero) would suggest the opposite.

In Figure 27, we reproduce an experiment from Campos *et al.* (2016), where a scatter-plot of the diversity and difficulty scores for several downsampled variants of a variety of base datasets is displayed. For each dataset with different downsampled variants, the 95% confidence ellipse is shown as well as the individual points for each variant. The means of the ellipses are indicated by an “x”, labeled with the corresponding base dataset. The datasets with only one variant are shown as a filled square. Unlike the original plot (CAMPOS *et al.*, 2016), our plot in Figure 27 maps the value of the adjusted IREOS for each dataset into a cold/hot color. Notice that datasets with high difficulty and diversity scores do tend to exhibit lower values of IREOS. Since these two strategies to assess the hardness of a dataset are based on completely different evaluation mechanisms, their compatible results suggest that both can be useful and that the intuition of separability adopted by IREOS seems to capture well the (not always explicit) intuition of the various outlier detection methods used in the study of Campos *et al.* (2016).

Supported by these results, and to increase the chances that the solutions of the unsupervised outlier detection methods are not completely meaningless with respect to the ground truth labels, we chose for each of the 23 dataset from Campos *et al.* (2016), the variant with the lowest

Figure 27 – Scatter-plot of *difficulty* versus *diversity* for several downsampled variants of a variety of base datasets, following Campos *et al.* (2016), but coloring points according to the IREOS assessment of the ground truth labels (adjusted for chance, with $m_{cl} = n$, where n is the number of outliers in the ground truth). Hotter colors represent *lower* IREOS values, *i.e.*, the ground truth does agree to a smaller degree with the notion of separability adopted by IREOS. Note that the IREOS score appears to be compatible with the difficulty score. The result obtained with $m_{cl} = 1$ is very similar.

difficulty score, whenever there are multiple downsampled variants of the dataset available for the chosen percentage of outliers. We chose from the variants with 5% of outliers, normalized, and without duplicates.

The collection of 23 real world datasets (CAMPOS *et al.*, 2016), some of which are displayed in Figure 27, does not include four of the 11 datasets used in the experiments in our preliminary publication (MARQUES *et al.*, 2015), namely, *Isolet*, *Multiple Features*, *Optical Digits*, and *Vowel*. For the sake of completeness, here we will also include these four datasets, resulting in 27 datasets in total.

5.4.2 Methods and Measures

In our experiments we evaluate results by contrasting the unsupervised assessment of candidate outlier solutions performed by IREOS against the supervised assessment using the

ground truth, *i.e.*, using the labels as provided in the datasets.⁵ We then study the relationship between the quality assessments of the solutions with respect to the ground truth and the quality assessments of the solutions computed by IREOS. To assess the quality of a given scoring solution with respect to the ground truth, *i.e.*, in a supervised way, we compute the Area Under the ROC Curve (ROC AUC) of the outlier scores in the solution under evaluation against the labels in the ground truth. For the real world datasets, the ground truth labels are binary (*i.e.* outlier/inlier, according to a certain classification perspective), but this is not necessarily the case for the synthetic datasets, where the true probability distributions are known and p -values (or rather their complement) can be used as reference outlier probabilities when computing the ROC AUC.

In order to produce a diverse collection of candidate outlier solutions to be assessed, we collected 10 different solutions for each dataset. For the *real world datasets*, these solutions were produced by the same outlier detection algorithms involved in the benchmarking study of Campos *et al.* (2016), namely: COF (TANG *et al.*, 2002), FastABOD (KRIEGEL; SCHUBERT; ZIMEK, 2008), INFLO (JIN *et al.*, 2006), KDEOS (SCHUBERT; ZIMEK; KRIEGEL, 2014a), KNN (RAMASWAMY; RASTOGI; SHIM, 2000), KNNW (ANGIULLI; PIZZUTI, 2002; ANGIULLI; PIZZUTI, 2005), LDOF (ZHANG; HUTTER; JIN, 2009), LDF (LATECKI; LAZAREVIC; POKRAJAC, 2007), LOF (BREUNIG *et al.*, 2000), LoOP (KRIEGEL *et al.*, 2009), ODIN (HAUTAMÄKI; KÄRKKÄINEN; FRÄNTI, 2004), and SimplifiedLOF (SCHUBERT; ZIMEK; KRIEGEL, 2014b). In addition to these algorithms, we also included a recent outlier detection method called GLOSH (CAMPELLO *et al.*, 2015), for a total of 13 algorithms. All these algorithms share a parameterization in terms of a local neighborhood size, hereafter referred to as k , which is the main parameter in all of these algorithms (and the only parameter in many of them). For those algorithms that have additional parameters, following Campos *et al.* (2016), we used the default values suggested by the original authors. For each real dataset, the algorithms were run by varying their neighborhood size k between 1 and 100.⁶ In order to select a diverse subset of 10 candidate solutions for each dataset, we ensured that the best and the worst solutions according to their ROC AUC were selected, and the remaining solutions were selected trying to keep an interval as equally spaced as possible between solutions in terms of their ROC AUC values.

For the *synthetic datasets*, since the actual distributions used to generate the data are known, algorithms are not needed to produce suitable candidate solutions. Instead, the candidate solutions were produced artificially, in a fully controlled way. We start from the perfect solution given by the ground truth (ROC AUC = 1), which is as follows: based on the known covariance matrix of the multivariate normal distribution that generated each data object, the Mahalanobis

⁵ Notice that these labels are not used by our index in any way, except if the index is specifically applied to assess the ground truth itself as a candidate solution, as in Figure 27.

⁶ Notice that, except for GLOSH, all these results have been precomputed and are available in the repository associated with the benchmarking study of Campos *et al.* (2016).

distance between each object and the mean of the distribution is computed. The distribution of the Mahalanobis distances follows a χ_d^2 distribution with d degrees of freedom, from which an outlier probability for each data object can be readily obtained as the complement of the p -value. For each synthetic dataset, the corresponding p -value complements are taken as a non-binary ground truth, *i.e.*, a perfect scoring solution. In order to produce a diverse collection of 10 candidate outlier solutions to be assessed, for each dataset we start from the ideal solution as described and iteratively produce new solutions by swapping the outlier score of one outlier ($p\text{-value} < 0.025$) with the score of a random inlier. This way, at each new iteration the ROC AUC deteriorates as we are worsening the ideal solution by flipping the scores of outliers and inliers (as per a 97.5% confidence level). Once again, in order to select a diverse subset of 10 candidate solutions for each dataset, we ensured that the best and the worst solutions according to their ROC AUC were selected, and the remaining solutions were selected trying to keep an interval as equally spaced as possible between solutions in terms of their ROC AUC values.

It is worth noticing that the outlier scores in the candidate outlier solutions obtained for the synthetic datasets already can be interpreted as (complementary) outlier probabilities (p -values), so the normalization procedure used by IREOS to produce regularized weights from the raw outlier scores is not necessary. As for the real datasets, whose candidate solutions have been obtained from a diverse collection of algorithms producing outlier scores with different characteristics, the normalization is required. In these cases, we applied the Gaussian scaling setting of the normalization framework by [Kriegel *et al.* \(2011\)](#), which has been recommended by the authors based on the overall results presented in their paper.

Experiments

The experiment of this chapter is designed to assess the effectiveness of IREOS, which is tested in two different experimental scenarios. In the first scenario, we measure the goodness of fit between the rankings obtained by assessing the 10 candidate solutions of each dataset in a supervised way, using ROC AUC, and in an unsupervised way, using IREOS. The goodness of fit is measured by computing the Spearman correlation between these two rankings of the same 10 candidate solutions, for each dataset. The second scenario involves model selection: using the quality assessments of the solutions computed by IREOS, we selected the best solution of each dataset according to the IREOS score, this solution is then compared in terms of ROC AUC against the best, worst, and expected (average) ROC AUC values across the entire collection of candidate solutions.

Parameters

We evaluate IREOS with $m_{cl} = 1$ (no modeling of clumps) and $m_{cl} = n$, where n is the number of outliers according to the ground truth (for synthetic datasets, n is the number of

data objects with p -value < 0.025 , *i.e.*, around 2.5% of the dataset size). Since n is unknown in practical application scenarios, we also experiment with another (empirical and generic, not domain-specific or application-oriented) setting for the case where the optional mechanism for modeling clumps is used: $m_{cl} = \sqrt{5\% \cdot N}$.

Following our preliminary publication (MARQUES *et al.*, 2015), the number of discrete γ values for the computation of the index is $n_\gamma = 100$. As previously discussed, the constant penalty cost for soft margin violations, C , only needs to be large enough as to perform its intended role, and our experiments show that the results obtained with values of C varying several orders of magnitude are very similar and do not change the conclusions. For instance, we showed earlier (MARQUES *et al.*, 2015) that the differences in the results for C varying from 100 to 800.000 are negligible. The experiments reported in this thesis have been run with $C = 100$.

5.4.3 Results

The results for the first scenario involving the correlation between the rankings of candidate solutions as determined by IREOS and by the ground truth, are summarized in Table 2. For the real datasets, which are individual datasets, the entries denote the value of the Spearman correlation between IREOS scores and ROC AUC values for the corresponding set of 10 candidate solutions. For the collection of synthetic data, the entries denote the average and the standard deviation of the Spearman correlation between IREOS scores and ROC AUC values over the multiple datasets of the collection.

Table 2 – Spearman correlation between IREOS (unsupervised) and ROC AUC (supervised).

Dataset	$m_{cl} = 1$	$m_{cl} = \sqrt{5\% \cdot N}$	$m_{cl} = n$
Annthyroid	0.248	-0.285	-0.394
Arrhythmia	0.539	0.673	0.721
Cardiotocography	0.758	0.915	0.976
Glass	0.467	0.479	0.382
HeartDisease	0.927	0.903	0.903
Hepatitis	0.939	0.903	0.903
InternetAds	0.636	0.915	0.782
Ionosphere	0.648	0.721	0.709
Isolet	0.273	0.6	0.6
Lymphography	0.661	0.758	0.758
MultipleFeature	0.152	0.515	0.515
OpticalDigits	0.733	0.782	0.782
PageBlocks	0.927	0.964	0.952
Parkinson	0.515	0.515	-0.37
Pima	0.758	0.83	0.879
Shuttle	0.867	0.867	0.867
SpamBase	0.794	0.842	0.842
Stamps	0.588	0.636	0.685
Vowel	0.927	0.709	0.661
Waveform	0.818	0.855	0.891
WBC	0.042	0.042	0.042
WDBC	0.915	0.952	0.952
Wilt	-0.794	-0.903	-0.939
WPBC	0.479	0.648	0.806
Synthetic	1 ± 0	1 ± 0	1 ± 0

Notice that, for the synthetic data, the Spearman correlation is maximum across the entire collection of 30 datasets. It shows that the ordering of the candidate solutions based on the notion of outlierness as captured by IREOS is perfectly aligned with the ordering of candidate solutions based the notion of outlierness according to the underlying statistical mechanism that generated the data, irrespective of m_{cl} .

For the real world datasets, which represent a more intricate evaluation scenario where the ground truth labels may be fully or partially inconsistent with the spatial distribution of the data, as discussed in section 5.4.1, there is a clear correlation most of the time. However, low or even negative values of correlation can be observed for datasets that appear to have inconsistent labellings according to the measures of diversity and difficulty proposed by Campos *et al.* (2016), *e.g.*, Annthyroid and Wilt (see Figure 27). The Wilt database, in particular, showed a strong negative correlation. The PCA visualization of the 5-dimensional Wilt database, using the first 2 principal components, shown in Figure 28, can help explain this result. As can be seen, most of the outliers, plotted in red, lie deep in the denser region of the cluster, which can explain the negative correlation with IREOS, since most objects labeled as outliers in the ground truth seem to be clear inliers from the viewpoint of the spatial distribution of the data.

Figure 28 – PCA visualization of the Wilt dataset using the first 2 principal components. Outliers according to the ground truth are displayed in red.

When comparing the results for different values of clump size, the correlations in Table 2 are similar across different m_{cl} values for most datasets, which suggests that this optional parameter is not critical. As previously discussed, whether clumps should be modeled or not, and, if so, what the expectations about the maximum size of a clump (as opposed to a cluster) should be, are problem-specific decisions. With our three general settings for m_{cl} across all the datasets, we notice that, while for most datasets the results are similar, for certain datasets, such as Isolet, Multiple Features, and WPBC, modeling clumps ($m_{cl} > 1$) provides better results, whereas the opposite occurs for a few other datasets, most noticeably Parkinson and Vowel. Overall, our generic (*i.e.*, blind, not application-specific) heuristic $m_{cl} = \sqrt{5\% \cdot N}$ seems to provide a good compromise, if not the best result, in almost all cases. Comparisons of performances of IREOS for different values of m_{cl} must be taken with a grain of salt, though, because the ground

truth (*i.e.*, labels based on some sort of semantic) can be seen as only one of different possible perspectives of what outliers should be in each dataset.

The results for the experimental scenario involving model selection are summarized in Table 3, which shows the ROC AUC values corresponding to the worst, the expected (average), and the best candidate outlier solution for each dataset, along with the ROC AUC value of the solution selected as best according to IREOS (*i.e.*, in an unsupervised way). For each selected solution, the table also indicates which algorithm produced this solution. By using IREOS with $m_{cl} = n$, one would select the most accurate solution according to the ground truth in 10 out of the 24 datasets. IREOS with $m_{cl} = 1$ makes the best choice for 9 out of the 24. IREOS with $m_{cl} = \sqrt{5\% \cdot N}$ makes the best choice for 11 out of the 24. In cases where the best solution according to IREOS is not the one with highest ROC AUC value, the choice is often much better than the expected (average) and worst values, and often close to the quality of the best possible selection, for at least two of the three clump size settings used. Exceptions are the datasets Annthyroid, Wilt, and Glass. The Annthyroid and Wilt datasets were already expected to exhibit poor results due to the label inconsistency problem discussed in the previous experimental scenario and in section 5.4.1. The low value of ROC AUC in the best solution chosen by IREOS for the Glass dataset can be explained due to the large number of ties in the scorings produced by the solution. This solution produced only 8 non-null values of scorings for the 214 objects in the dataset. When the ROC curve is computed, past these 8 objects, the curve will walk diagonally, resembling a random solution, as shown in Figure 29a, which explains the low ROC AUC value. The PCA visualization of the Glass dataset for the first two principal components (Figure 29b) suggests that the 8 objects detected as outliers, plotted in red, are apparently not in clear disagreement with the spatial distribution of the data.

Figure 29 – Best solution selected by IREOS for the Glass dataset. Outliers are highlighted in red.

To get a better sense of where the selected solutions are located within the distribution of the ROC AUC values for all candidate solutions, we also show box plots of the distributions for each dataset in Figure 30. The position of the solutions selected by IREOS are indicated by special symbols in the plots.

Table 3 – Summarization of the ROC AUC values for all candidate solutions used for model selection. Three last columns indicate the solutions selected by IREOS for $m_{cl} = 1$, $m_{cl} = \sqrt{5\% \cdot N}$ and $m_{cl} = n$, respectively.

Dataset	Min	Max	Avg	IREOS ($m_{cl} = 1$)	IREOS ($m_{cl} = \sqrt{5\% \cdot N}$)	IREOS ($m_{cl} = n$)
Anthyroid	0.528	0.732	0.63	0.642 KNNW	0.574 KNN	0.551 KNN
Arrhythmia	0.5	0.916	0.737	0.879 GLOSH	0.879 GLOSH	0.879 GLOSH
Cardiotocography	0.536	0.839	0.689	0.74 LOF	0.74 LOF	0.839 LOF
Glass	0.492	0.904	0.699	0.539 GLOSH	0.539 GLOSH	0.539 GLOSH
HeartDisease	0.387	0.97	0.687	0.847 KNNW	0.847 KNNW	0.847 KNNW
Hepatitis	0.075	0.96	0.522	0.96 COF	0.96 COF	0.96 COF
InternetAds	0.413	0.895	0.656	0.843 FastABOD	0.843 FastABOD	0.843 FastABOD
Ionosphere	0.514	0.96	0.745	0.868 LoOP	0.868 LoOP	0.868 LoOP
Isolet	0.116	1	0.569	0.116 GLOSH	1 COF	1 COF
Lymphography	0.43	1	0.748	0.95 LDOF	0.95 LDOF	0.95 LDOF
MultipleFeatures	0.233	0.994	0.638	0.918 LOF	0.994 COF	0.994 COF
OpticalDigits	0.164	0.999	0.596	0.999 KNN	0.737 LDF	0.737 LDF
PageBlocks	0.506	0.943	0.728	0.896 LOF	0.943 LDF	0.943 LDF
Parkinson	0.448	1	0.765	1 FastABOD	1 FastABOD	0.51 LDF
Pima	0.499	0.834	0.67	0.762 LOF	0.762 LOF	0.762 LOF
Shuttle	0.415	0.992	0.708	0.992 LDF	0.87 KNNW	0.87 KNNW
SpamBase	0.463	0.779	0.625	0.779 KNNW	0.779 KNNW	0.779 KNNW
Stamps	0.38	0.921	0.655	0.921 KNN	0.921 KNN	0.921 KNN
Vowel	0.223	1	0.614	1 FastABOD	1 FastABOD	0.916 COF
Waveform	0.5	0.796	0.66	0.708 INFLO	0.796 LDF	0.796 LDF
WBC	0.438	0.997	0.721	0.997 GLOSH	0.997 GLOSH	0.997 GLOSH
WDBC	0.497	0.988	0.745	0.988 GLOSH	0.988 GLOSH	0.988 GLOSH
Wilt	0.34	0.713	0.527	0.424 KNNW	0.424 KNNW	0.382 KNNW
WPBC	0.401	0.583	0.493	0.543 LDF	0.543 LDF	0.543 LDF

Figure 30 – Distribution of the ROC AUC values for all candidate solutions used in the experiments. The position of the solutions selected by IREOS as best are indicated by special symbols displayed in red.

5.5 Chapter Remarks

We addressed in this chapter the long-term open problem (ZIMEK; CAMPELLO; SANDER, 2013) of internal and relative evaluation of outlier detection results, that is, the assessment of the quality of results of unsupervised outlier detection methods without referring to external information (such as class labels). In the typical application scenario of outlier detection, such external information about what the outliers are is not available in advance, and results need to be assessed by domain experts.

IREOS is the first measure to allow such quality assessment of solutions automatically and, as a consequence, to select better solutions (models, parametrizations) for a given problem. We discussed the properties of IREOS, including derived statistics, p -values, and adjustment for chance. We show the usefulness of IREOS for unsupervised quality assessment of outlier detection results in an experimental scenario involving the correlation between the rankings of candidate solutions as determined by IREOS and by the ground truth, as well as in another scenario involving practical experiments of model selection.

In these two different experimental scenarios, we evaluated IREOS using the optional parameter m_{cl} , which allows the domain expert to adjust the notion of clump size versus cluster size, in three different settings. We set $m_{cl} = 1$ to represent cases where clumps are not modeled, $m_{cl} = n$ to represent cases where the clumps are modeled with domain information, and $m_{cl} = \sqrt{5\% \cdot N}$ to represent cases where the clumps are modeled in the absence of domain information. When comparing the results for different values of clump size, the results are similar across different m_{cl} values for most datasets, which suggests that this optional parameter is not critical. However, we notice that, while for most datasets the results are similar, modeling clumps ($m_{cl} > 1$) tended to provide better results overall. Therefore, in absence of domain information to decide whether clumps should be modeled or not, or what should be the maximum size of a clump, we recommend our generic (i.e., not application specific/informed) heuristic $m_{cl} = \sqrt{5\% \cdot N}$, which has systematically provided a good compromise, if not the best result, in almost all cases.

A recurrent problem mentioned in the literature regarding IREOS is its high computational burden for the computation of the index (MARQUES, 2015; CAMPOS *et al.*, 2016; PANG *et al.*, 2017). The extended version presented in this chapter cannot rely on the same assumption as the original IREOS to be computed efficiently, which makes it computationally more challenging. In the next chapter, we introduce many different ways to mitigate the complexity of the index.

SPEEDUP TECHNIQUES FOR IREOS

In contrast to the original IREOS (MARQUES *et al.*, 2015; MARQUES, 2015), which can be computed efficiently by taking advantage of the imbalanced nature of the binary outlier detection problem, where the number of outliers, n , is given as input and is typically much smaller than the data set size, N , the extended version presented in this thesis (chapter 5) cannot rely on this assumption when evaluating non-binary solutions, which makes it computationally more challenging.

In this chapter, we provide strategies to avoid large amounts of unnecessary computations in the three major components involved in the complexity of the index. In section 6.1, we provide a strategy to efficiently compute the n_γ separability values across different values of γ for a given object. In section 6.2, we discuss the strategy to avoid unnecessary computations in the separability values across N different objects for a given value of γ . In section 6.3, we discuss the strategy to mitigate the computational burden to train each classifier for a dataset of size N . Also, we experimentally evaluate the efficiency and efficacy of the strategies proposed in section 6.4 and conclude the chapter in section 6.5.

6.1 Separability Curves

In practice, in order to calculate the separability curve for a given data object, we need to discretize the continuous range of γ values, $[0, \gamma_{max}]$, into n_γ discrete values. In the experiments performed in the previous chapter, we arbitrarily set $n_\gamma = 100$ as a setting that empirically provides a very accurate approximation of the AUC of separability as discretized into n_γ equally spaced values. There are, however, two major limitations with this approach. First, it does not provide any control of the trade-off between the resulting level of approximation (which is not known in advance) and the corresponding computational burden. The only information available is that, by increasing n_γ , the approximation (*i.e.* discretization) error will decrease at the price of

an increase in the computational cost, but the user cannot calibrate the desired approximation error and automatically derive the smallest n_γ that provides that error. It is possible, for instance, that a very similar (and already good enough) approximation can be achieved with much fewer than $n_\gamma = 100$ values, in which case this arbitrary setting will waste a large amount of computing power to unnecessarily and only slightly improve the quality of the approximation. In addition, an approach that discretizes values evenly into equally spaced intervals may not be the most effective and efficient. Instead, a smaller number of γ values can be sampled from less critical regions of the separability curve, particularly when the curve has already flattened and therefore requires much fewer values for a good approximation than required in the leftmost part of the curve.

To overcome these limitations and to significantly speedup the computation of IREOS, instead of using a fixed, arbitrary number n_γ , we will use a variable number n_γ according to a user-specified tolerance for an estimated approximation error. To that end, we make use of the technique of adaptive quadrature (or adaptive numerical integration) (KUNCIR, 1962; GONNET, 2012; BURDEN; FAIRES; BURDEN, 2016). This technique will approximate the separability curve by subdividing it recursively into smaller subintervals. The procedure begins with the approximation of the separability curve with the minimum possible number of points (three). If this approximation does not reach the user-specified precision, the separability curve is subdivided into two parts (bisection), and the same procedure is applied recursively and independently on both (equally spaced) intervals, until the user-specified precision is reached. This procedure is illustrated in Figure 31, where the three different separability curves in Figure 25b, corresponding to a global outlier, a local outlier, and an inlier, are approximated by this procedure. Initially, in Figure 31a, the separability curve represented by the solid line is approximated by the dotted line using only three points. Since the approximation is not good enough, the separability curve is divided into two parts, the area of these two parts are calculated independently, and the procedure is applied recursively to both of them. Figure 31b illustrates the bisection as a vertical line as well as the approximated curve on both sides (five points in total) as a dotted line. Since the area on the right side has already reached a satisfactory approximation, only the left part will be further subdivided (Figure 31c). This procedure is repeated recursively, until each subdivision reaches a sufficiently good approximation (Figure 31d). The other two separability curves are approximated using the same procedure, in Figures 31e-31g and 31h, respectively. Note that the number of points needed to achieve a good approximation varies in each case.

(a) Approximation using 3 points (global outlier)

(b) Approximation using 5 points (global outlier)

(c) Approximation using 7 points (global outlier)

(d) Approximation using 21 points (global outlier)

(e) Approximation using 3 points (local outlier)

(f) Approximation using 5 points (local outlier)

(g) Approximation using 9 points (local outlier)

(h) Approximation using 3 points (inlier)

Figure 31 – Approximation of the separability curves in Figure 25b using the adaptive quadrature technique. The original curves are shown as solid lines, whereas the approximate curves are shown as dotted lines. Bisections are highlighted as vertical lines.

The stopping condition of this procedure is that the approximation error should be less than a user-specified threshold. Since the exact error cannot be determined as it would require the exact computation of the separability curves, which is what the use of an approximation is meant to avoid in the first place, an estimated error bound can be used instead. Since IREOS is defined conceptually as the AUC of the continuous average curve of separability $\bar{p}(\gamma)$, which we want to approximate by a finite number of points within the interval $[0, \gamma_{\max}]$, determined using the adaptive quadrature technique, the error bound for this approximation can be calculated analytically through a closed-formula (BURDEN; FAIRES; BURDEN, 2016). In particular, there is a class of error bound, known as *a posteriori* error estimate, that gives us a computable estimate based on the computed approximation only.

There are different approaches to calculate this type of error estimate (GONNET, 2012). The most commonly used approach is based on the difference between the approximation of the curve over an entire interval, using n_γ points, and the approximation of the curve after a bisection of the original interval, as the sum of the approximations over the two resulting subintervals, *i.e.*, using $2n_\gamma$ points (although the midpoint is shared by both intervals, it is technically used twice, once in each interval).

Let $I(0, \gamma_{\max})$ be the approximate IREOS index computed over the entire interval $[0, \gamma_{\max}]$ as in Equation (5.9), *i.e.*, using n_γ discrete points. In addition, let $I(0, \frac{\gamma_{\max}}{2})$ and $I(\frac{\gamma_{\max}}{2}, \gamma_{\max})$ be the approximate IREOS index computed separately in each bisection subinterval, $[0, \frac{\gamma_{\max}}{2}]$ and $[\frac{\gamma_{\max}}{2}, \gamma_{\max}]$, respectively, each of which uses n_γ points. Since IREOS is, by definition, an AUC, it follows that $I(0, \frac{\gamma_{\max}}{2}) + I(\frac{\gamma_{\max}}{2}, \gamma_{\max})$ corresponds to the approximate IREOS index computed over the entire interval $[0, \gamma_{\max}]$ after the bisection, *i.e.*, using $2n_\gamma$ points, in contrast to $I(0, \gamma_{\max})$, which uses n_γ points only (*i.e.*, before the bisection). In addition, let I_{exact} be the exact value of the index as if the exact (continuous) curves of separability were known. For smooth and continuous curves, such as separability curves, we can estimate the error of the approximation using $2n_\gamma$ points as (DAVIS; RABINOWITZ, 1984; ASCHER; GREIF, 2011; GONNET, 2012):

$$\begin{aligned} \mathcal{E}_{2n_\gamma}(I) &:= |I_{\text{exact}} - (I(0, \frac{\gamma_{\max}}{2}) + I(\frac{\gamma_{\max}}{2}, \gamma_{\max}))| \\ &\approx \frac{1}{3} | (I(0, \frac{\gamma_{\max}}{2}) + I(\frac{\gamma_{\max}}{2}, \gamma_{\max})) - I(0, \gamma_{\max}) | \end{aligned} \quad (6.1)$$

Equation (6.1) can be interpreted as follows: the absolute error between the exact curve and its (bisected) approximation using $2n_\gamma$ points, $\mathcal{E}_{2n_\gamma}(I)$ (the first expression above, by definition), is approximately $\frac{1}{3}$ of the absolute difference between the approximation using $2n_\gamma$ points and the (non-bisected) approximation using n_γ points only. Notice that, while the former expression is not computable because we do not know I_{exact} , the components in the latter expression are not only computable but also readily available as we perform the adaptive quadrature procedure. After each new bisection step is performed, we can recursively apply Equation (6.1) to the bisected subinterval, update the total estimated error (as a sum of the errors in all current subintervals), until this error falls below the user-defined threshold, when the adaptive quadrature can be interrupted.

It is worth noticing that, for the sake of simplicity, the above description of the adaptive quadrature procedure has subsumed the use of the midpoint rule as the method for numerical integration, but there are other rules that can be applied. In case of higher order methods, the ratio between the error when using $2n_\gamma$ points and the error when using only n_γ points can be much more accentuated than $\frac{1}{3}$. For instance, for the Simpson's rule, which is used in our code and experiments, the ratio is $\frac{1}{15}$ (BURDEN; FAIRES; BURDEN, 2016). This means that the estimated error decays much more quickly and, therefore, a smaller number of points is required to achieve a given targeted approximation quality.

6.2 Separability Values Across the Dataset

Due to the normalization applied to the outlier scores, following the approach of Kriegel *et al.* (2011) to compute weights w_j as outlier probabilities associated with the data objects $\mathbf{x}_j$, many objects in the data set will be assigned a weight of zero. Since the separability $p(\mathbf{x}_j, \gamma)$ of each object $\mathbf{x}_j$ is multiplied by the corresponding weight w_j in Equation (5.9), the separabilities of all objects with weight equal to 0 do not need to be computed, which in practice can drastically reduce the number of classifiers to be trained.

The normalization proposed by Kriegel *et al.* (2011) presumes the regularization of the scores in such a way that the expected score for an object considered to be an inlier by the outlier detection method that produced the scores will have an associated probability (weight) around zero. A number of outlier detection algorithms in the literature (BREUNIG *et al.*, 2000; ZHANG; HUTTER; JIN, 2009; PAPADIMITRIOU *et al.*, 2003; CAMPELLO *et al.*, 2015) readily provide an expected score as a mathematical property, so that regularization can be done just by subtracting this value from the outlier scores — *e.g.*, the expected score for an inlier is around 1 for LOF (BREUNIG *et al.*, 2000) and around 0 for GLOSH (CAMPELLO *et al.*, 2015). In case of algorithms for which the expected score of inliers cannot be trivially determined, the regularization must be performed based on assumptions about the distribution of the outlier scores; *e.g.*, if a Normal distribution is assumed, the regularization can be performed by subtracting the mean or median from the scorings, “squeezing” the scores of objects at or below the mean/median to zero. Notice that, in practice, for algorithms whose expected score for inliers is known, *e.g.* LOF, a large amount of the data objects, as inliers, will tend to have null weight, and therefore it is not unrealistic to assume that, even in the *full scoring evaluation* scenario, the separability values and the corresponding classifiers may only need to be computed for a fraction n of the data objects, where $n \ll N$.

In practice, one of the most important applications of IREOS could be *model selection*, where the analyst faces a collection of candidate outlier detection solutions produced by different algorithms, or different parameters of the same algorithm, and wants to relatively compare and rank these solutions, in a completely unsupervised way, in order to pick the best one(s) for

further investigation. In this scenario, it is possible to further speed-up computations by taking advantage of a *pruning strategy*, explained below.

Since larger weights $w_{(\cdot)}$ have greater influence on the composition of the index in Equation (5.9), computing the separability values only for a subset of the data objects with the largest weights may suffice to determine that one solution is necessarily better or worse than others. By sorting the weights in each candidate solution and then computing IREOS incrementally by processing the normalized products $(p(\mathbf{x}_j, \gamma) \cdot w_j) / (\sum_i w_i)$ in decreasing order of weights w_j , simultaneously across all candidate solutions, we can monitor the current partial value of the index (I_{partial}) for each solution as well as the maximum value that the index can reach for that solution ($\hat{I}_{\text{max}}$) assuming that all the data objects that have not yet been processed would have maximum separability of 1. Since both the separability as well as the weight values cannot be negative, *i.e.*, $p(\mathbf{x}_j, \gamma) \geq 0$ and $w_j \geq 0$, I_{partial} can only increase as we process more objects. Contrarily, $\hat{I}_{\text{max}}$ can only decrease. Hence, by keeping track and comparing the values of I_{partial} of each solution against the values of $\hat{I}_{\text{max}}$ for the other solutions, we can stop the calculation of the index of a solution earlier, when it cannot surpass or be surpassed by any other solution.

An example of this pruning strategy is illustrated in Table 4, where three outlier detection solutions are evaluated. The outlier weights are sorted in decreasing order and the separability of the corresponding data objects are computed incrementally (from top to bottom). In each iteration, one data object is processed at each candidate solution, and the values of I_{partial} and $\hat{I}_{\text{max}}$ for that solution are updated and compared with those of the other solutions. Once I_{partial} of a solution is already larger than the $\hat{I}_{\text{max}}$ values of all the other solutions that are still active (*i.e.*, being processed), the process of this solution can be interrupted. In the example in Table 4, this occurs in the 4th iteration, when Solution 1 no longer needs to be processed as its partial index ($I_{\text{partial}} = 0.739$) cannot be reached by the indexes in Solutions 2 ($\hat{I}_{\text{max}} = 0.686$) and 3 ($\hat{I}_{\text{max}} = 0.543$). We can also halt processing a solution once its $\hat{I}_{\text{max}}$ value is already smaller than I_{partial} of all the other solutions that are still active. In our example, this occurs in the 7th iteration, when Solution 3 no longer needs to be processed as its maximum possible index value ($\hat{I}_{\text{max}} = 0.5007$) cannot reach the index in Solution 2 ($I_{\text{partial}} = 0.5095$).¹

6.3 Classifiers and Computational Complexity Revisited

The computational cost of IREOS is dominated by the cost of training multiple classifiers. While maximum margin classifiers are known for their high efficacy, they are also known for

¹ There is also a 3rd scenario in which we can stop processing a solution: let Ω be the set of candidate solutions. If $\hat{I}_{\text{max}}$ of a solution ω_i is already lower than the $\hat{I}_{\text{partial}}$ values of a subset of solutions $\Omega_A \subset \Omega$, while $\hat{I}_{\text{partial}}$ of ω_i is already higher than the $\hat{I}_{\text{max}}$ values of the remaining solutions, *i.e.* those in the subset $\Omega \setminus \{\Omega_A, \omega_i\}$, then ω_i cannot surpass or be surpassed by any other solution and can be stopped.

Table 4 – Illustrative example of the pruning strategy for model selection involving three hypothetical candidate outlier detection solutions. Processing of Solution 1 can be stopped earlier, at the 4th iteration, since it cannot be surpassed by Solutions 2 or 3. Processing of Solution 3 can be stopped at the 7th iteration, as it cannot surpass Solutions 1 or 2.

Solution 1				Solution 2				Solution 3			
$p(\mathbf{x}_i, \gamma)$	w_i	I_{partial}	$\hat{I}_{\text{max}}$	$p(\mathbf{x}_j, \gamma)$	w_j	I_{partial}	$\hat{I}_{\text{max}}$	$p(\mathbf{x}_l, \gamma)$	w_l	I_{partial}	$\hat{I}_{\text{max}}$
1	0.3	0.3	1	0.2	0.3	0.06	0.76	0.6	0.35	0.21	0.86
0.87	0.2	0.474	0.974	0.6	0.15	0.15	0.7	0.2	0.25	0.26	0.66
1	0.135	0.609	0.974	0.9	0.14	0.276	0.686	0.5	0.15	0.335	0.585
1	0.13	0.739	0.974	1	0.1	0.376	0.686	0.4	0.07	0.363	0.543
0.3	0.12	0.775	0.89	0.95	0.09	0.4615	0.6815	0.3	0.05	0.378	0.508
0.6	0.1	0.835	0.85	0.4	0.08	0.4935	0.6335	0.9	0.045	0.4185	0.5035
0.2	0.015	0.838	0.838	0.2	0.08	0.5095	0.5695	0.92	0.035	0.4507	0.5007
0.3	0	0.838	0.838	1	0.06	0.5695	0.5695	1	0.03	0.4807	0.5007
0.5	0	0.838	0.838	0.1	0	0.5695	0.5695	1	0.02	0.5007	0.5007

being computationally demanding. Fortunately, different techniques are available in the literature to lessen the computational burden of these classifiers. These techniques can be categorized into two main approaches: *algorithmic* and *data selection*. The first approach consists of faster algorithms for the Quadratic Programming (QP) solver required by maximum margin classifiers (PLATT, 1998; KEERTHI *et al.*, 2005), while the latter approach focuses on the selection of training data in order to reduce the number of examples (N) used to train the classifiers (LEE; MANGASARIAN, 2001; PANDA; CHANG; WU, 2006). Our approach to speeding up the computation of our index combines the best of both worlds: while taking advantage of a fast QP solver (KEERTHI *et al.*, 2005), we also elaborate on a suitable approach for selection of the training data.

The most straightforward approach to reduce N is to randomly subsample the dataset available for training (LEE; MANGASARIAN, 2001). However, when evaluating the individual separability of a data object, an informed subsampling is required instead, because separability can be highly affected depending on the objects filtered out by the sampling procedure. In Figure 32, we can see two different subsamples taken from the dataset illustrated in Figure 32a. Notice that, when evaluating the separability of the object marked as a red square, while the subsample in Figure 32b completely modified the decision boundary by filtering out a single object, the boundary in Figure 32c has not been affected even though an entire cluster has disappeared. While the use of soft margin classifiers (as opposed to the strict margin illustrated in Figure 32) significantly reduces the influence of single data objects on the computation of separability, the message learned from the extreme case illustrated in Figure 32 is still valid: closer objects have more influence on the decision boundary and, accordingly, on the separability of a given object.

While closer neighbors noticeably affect the decision boundary for the separability of a given object, other more distant objects are likely to be irrelevant. In fact, for certain classifiers, such as SVMs, only the support vectors matter. For Kernel Logistic Regression (KLR), while all objects influence the decision boundary to some extent, the influence tends to zero very

Figure 32 – Illustrative example of subsampling and the influence of close and distant objects on an object's separability.

quickly as an object moves away from the margin boundary on the correct side. Thus, in order to select the most effective training subset for our classifiers, we follow previous work from the literature (PANDA; CHANG; WU, 2006; MICENKOVÁ *et al.*, 2013) and propose to use only the k Nearest Neighbors (kNN) of each object (rather than the whole dataset) to evaluate its separability.

When trained from the N objects of a dataset to compute the separability of one particular object, the KLR classifier, which is used in all our experiments, runs in $\mathcal{O}(d \cdot N^3)$ time, where d is the data dimensionality. By using a subsample containing only the k nearest neighbors of the object in question, this complexity is drastically reduced to $\mathcal{O}(d \cdot k^3)$, plus $\mathcal{O}(N)$ to determine the k nearest neighbors using an ordinary linear search, *i.e.*, $\mathcal{O}(d \cdot k^3 + N)$. Then, in the worst case scenario where, despite the different pruning strategies previously described, the separability (and the k nearest neighbors) of all the N objects in the data set would be required to compute IREOS, for n_γ different values of γ , the total complexity of the index would be $\mathcal{O}(N \cdot n_\gamma \cdot d \cdot k^3 + N^2)$.

If appropriate index structures are in place, such as, *e.g.*, a Kd-Tree, the k nearest neighbors for all N observations can be determined in $\mathcal{O}(N \log N)$ time. Moreover, notice that the exact k nearest neighbors are not fundamentally important, because, by using a subsample, we are already computing an approximate value of separability in the first place, so an approximate-NN approach (GIONIS; INDYK; MOTWANI, 1999; SCHUBERT; ZIMEK; KRIEGEL, 2015) should suffice. By using approximate rather than exact nearest neighbors, the k (approximate) nearest neighbors for all N observations may be determined in $\mathcal{O}(N)$ time, typically with the aid of hashing structures. In this case, the complexity of IREOS in the worst case reduces to $\mathcal{O}(N \cdot n_\gamma \cdot d \cdot k^3 + N)$, *i.e.*, $\mathcal{O}(N \cdot n_\gamma \cdot d \cdot k^3)$. For top- n solutions, the complexity is $\mathcal{O}(n \cdot n_\gamma \cdot d \cdot k^3)$.

The value of k is a compromise between the quality of approximation and runtime. It is important to notice that most existing unsupervised outlier detection algorithms require the computation of the k nearest neighbors for all observations in the data set, where k is a user-defined parameter of these algorithms (typically a small constant such that $k \ll N$). If the same value of k given as input to the algorithm or algorithms that produced the outlier detection

solution(s) to be evaluated is also used to compute IREOS, the k nearest neighbors computed by the algorithms can be reused with no extra cost to IREOS. This makes the worst-case complexity of IREOS to be $\mathcal{O}(N \cdot n_\gamma \cdot d \cdot k^3)$ for scoring solutions or $\mathcal{O}(n \cdot n_\gamma \cdot d \cdot k^3)$ for top- n solutions even when using exact (rather than approximate) nearest neighbors. Finally, notice that these results subsume centralized processing, but one can significantly speed up computations using parallelization schemes (as we do in our code).

6.4 Evaluation

The experiments of this chapter are designed to assess the compromise between computational efficiency and efficacy of the strategies proposed to speed up computation of IREOS. First, each proposed strategy is evaluated separately in terms of computational gain and loss of approximation quality in relation to the original index. Then, all the proposed strategies are combined and compared against the original index. Finally, the experiments of the previous chapter are repeated using these combined strategies.

6.4.1 Efficiency of Approximations

Table 5 – Comparison between IREOS with fixed n_γ and variable n_γ as determined by the adaptive quadrature approximation.

Dataset	Fixed n_γ	Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.01$		Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.005$		Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.001$	
	Classifiers	Classifiers	Diff	Classifiers	Diff	Classifiers	Diff
Annthyroid	694200	41704.2 (6.01%)	0.035	62081.2 (8.94%)	0.00155	76952.8 (11.09%)	0.00036
Arrhythmia	25600	1536 (6%)	0.04594	5684.2 (22.2%)	0.00347	8143.2 (31.81%)	0.00075
Cardiotocography	173400	10614.8 (6.12%)	0.01484	16917.4 (9.76%)	0.01709	27562.2 (15.9%)	0.01543
Glass	21400	1288.8 (6.02%)	0.05718	3492.6 (16.32%)	0.00173	4996.8 (23.35%)	0.00031
HeartDisease	15700	942 (6%)	0.03257	2162.2 (13.77%)	0.00466	3368 (21.45%)	0.00098
Hepatitis	7000	420 (6%)	0.04733	1354.4 (19.35%)	0.0033	1889.8 (27%)	0.00057
InternetAds	168200	10341 (6.15%)	0.02633	28822.2 (17.14%)	0.00705	44124.4 (26.23%)	0.00521
Ionosphere	35100	2137.8 (6.09%)	0.04363	6925.8 (19.73%)	0.00922	9435.4 (26.88%)	0.00835
Isolet	91000	5460 (6%)	0.02557	18553 (20.39%)	0.0052	26693.2 (29.33%)	0.00089
Lymphography	14800	888 (6%)	0.03175	2108.4 (14.25%)	0.00453	3246.2 (21.93%)	0.00105
MultipleFeature	40800	2448 (6%)	0.03655	8070.8 (19.78%)	0.00529	12005.6 (29.43%)	0.00107
OpticalDigits	114400	6889.8 (6.02%)	0.01573	17985.8 (15.72%)	0.00786	28545 (24.95%)	0.00146
PageBlocks	513900	30919.4 (6.02%)	0.01231	39310.6 (7.65%)	0.00287	48580.8 (9.45%)	0.00067
Parkinson	5000	300 (6%)	0.04883	888 (17.76%)	0.00222	1303.6 (26.07%)	0.00048
Pima	52600	3225.4 (6.13%)	0.02824	7307.2 (13.89%)	0.00768	11381.2 (21.64%)	0.0062
Shuttle	101300	6078 (6%)	0.01641	7087.2 (7%)	0.0155	8810.2 (8.7%)	0.01529
SpamBase	266100	16701.4 (6.28%)	0.02191	43310.2 (16.28%)	0.00502	62923 (23.65%)	0.001
Stamps	32500	1960.6 (6.03%)	0.02733	3916.2 (12.05%)	0.00939	6134.4 (18.88%)	0.00823
Vowel	10000	600 (6%)	0.0545	980 (9.8%)	0.05453	1576.6 (15.77%)	0.05357
Waveform	344300	21198.4 (6.16%)	0.00765	55365.6 (16.08%)	0.00754	86595.8 (25.15%)	0.00192
WBC	22300	1352.4 (6.06%)	0.05597	4047.4 (18.15%)	0.00382	5801 (26.01%)	0.00317
WDBC	36700	2231.4 (6.08%)	0.04083	6520.2 (17.77%)	0.00432	9673.8 (26.36%)	0.00209
Wilt	481900	28979.4 (6.01%)	0.01374	37752.4 (7.83%)	0.01388	48340.8 (10.03%)	0.01992
WPBC	19800	1188 (6%)	0.05295	3778.2 (19.08%)	0.00299	5555 (28.06%)	0.00048

In this experiments, we evaluate the fast, approximate strategies to compute IREOS discussed in this chapter. We start with a comparison between the computational cost of the index

when using a fixed number $n_\gamma = 100$ of points to compute the separability curves and the cost when using the adaptive quadrature approach. We experiment with the estimated approximation error set to 0.01, 0.005 and 0.001. The results are shown in Table 5, where columns with header “Classifiers” show the average number of classifiers trained to evaluate a solution, whereas columns with header “Diff” show the average absolute difference between the index as computed with $n_\gamma = 100$ and as computed with a variable n_γ determined by the adaptive quadrature technique. All the averages were taken over the assessment of 30 candidate solutions, 10 for $m_{cl} = 1$, 10 for $m_{cl} = \sqrt{5\% \cdot N}$, and 10 for $m_{cl} = n$. For the estimated error of 0.01, which was the largest one used, the number of classifiers needed to evaluate a solution with adaptive quadrature falls as low as 6% of the amount required with n_γ fixed, while the observed approximation errors (Diff) are in the second or third decimal digit. For an estimated error of 0.001, all datasets required less than a third of the classifiers needed with n_γ fixed, and most Diff values were in the third or fourth decimal digit.

Table 6 – Comparison between IREOS with and without pruning strategies.

Dataset	No pruning	Pruning: $w_{(\cdot)} = 0$	Pruning: Stop Early
	Classifiers	Classifiers	Classifiers
Annnthyroid	694200	232710 (33.52%)	193816.7 (27.92%)
Arrhythmia	25600	13820 (53.98%)	13753.33 (53.72%)
Cardiotocography	173400	69490 (40.07%)	66096.67 (38.12%)
Glass	21400	7480 (34.95%)	7010 (32.76%)
HeartDisease	15700	7250 (46.18%)	6866.667 (43.74%)
Hepatitis	7000	3320 (47.43%)	3236.667 (46.24%)
InternetAds	168200	79830 (47.46%)	78270 (46.53%)
Ionosphere	35100	21890 (62.36%)	20700 (58.97%)
Isolet	91000	45400 (49.89%)	44393.33 (48.78%)
Lymphography	14800	6480 (43.78%)	6210 (41.96%)
MultipleFeature	40800	19430 (47.62%)	19080 (46.76%)
OpticalDigits	114400	57420 (50.19%)	53643.33 (46.89%)
PageBlocks	513900	204170 (39.73%)	184410 (35.88%)
Parkinson	5000	2230 (44.6%)	2053.333 (41.07%)
Pima	52600	20930 (39.79%)	19926.67 (37.88%)
Shuttle	101300	47120 (46.52%)	42490 (41.94%)
SpamBase	266100	172240 (64.73%)	168766.7 (63.42%)
Stamps	32500	12820 (39.45%)	12406.67 (38.17%)
Vowel	10000	3310 (33.1%)	3060 (30.6%)
Waveform	344300	163570 (47.51%)	158636.7 (46.08%)
WBC	22300	8720 (39.1%)	8166.667 (36.62%)
WDBC	36700	15660 (42.67%)	14833.33 (40.42%)
Wilt	481900	171230 (35.53%)	148256.7 (30.77%)
WPBC	19800	8200 (41.41%)	7960 (40.2%)

Our next experiment assesses the reduction of the computational cost when avoiding the unnecessary computation of separability curves for data objects with zero outlier probability (weight $w_{(\cdot)} = 0$) as well as by using the pruning strategy that stops early when ranking candidate solutions for model selection. The results are shown in Table 6. Notice that, in this comparison scenario, Diff is irrelevant, because: (a) Pruning by $w_{(\cdot)} = 0$ produces an exact result (Diff = 0), not an approximation; and (b) in model selection, the exact value of the index is not important, only the ranking of the candidate solutions matters, and the stop early pruning strategy produces the same ranking of candidate solutions as the one that would be obtained by sorting the solutions according to the index computed without any pruning. Notice from Table 6 that each of these

Table 7 – Comparison between IREOS with and without the use of kNN approximation to compute separability. The reported runtimes are the average times to compute each classifier. The number of classifiers does not change in this strategy.

Dataset	Full Dataset	$k = 10$		$k = 50$		$k = 100$		$k = 250$	
	Runt. Class.	Diff	Runtime Class.	Diff	Runtime Class.	Diff	Runtime Class.	Diff	Runtime Class.
Cardiotocogr.	382ms	0.0565	0.07ms (0.02%)	0.0195	0.57ms (0.15%)	0.0172	1.69ms (0.44%)	0.0161	8.49ms (2.22%)
InternetAds	22s	0.0382	0.65ms ($\approx 0\%$)	0.0121	7.01ms (0.03%)	0.0102	30.18ms (0.13%)	0.0087	493.93ms (2.17%)
PageBlocks	3s	0.0908	0.04ms ($\approx 0\%$)	0.0418	0.38ms (0.01%)	0.0383	1.26ms (0.04%)	0.0294	6.89ms (0.2%)
SpamBase	1s	0.0369	0.07ms ($\approx 0\%$)	0.0292	0.69ms (0.04%)	0.0336	2.54ms (0.15%)	0.0319	11.12ms (0.66%)
Waveform	1s	0.0308	0.04ms ($\approx 0\%$)	0.0130	0.47ms (0.03%)	0.0078	1.58ms (0.08%)	0.0038	8.65ms (0.46%)
Wilt	2s	0.1105	0.04ms ($\approx 0\%$)	0.0510	0.34ms (0.01%)	0.0410	1.13ms (0.04%)	0.0266	6.20ms (0.23%)

pruning techniques is able to reduce computations to less than 50% of the original cost in most datasets.

Our third experiment assesses the trade-off between average runtime and quality of approximation of IREOS with the separability of each object computed using the object’s k nearest neighbors only, as opposed to using all the other objects in the dataset. We varied the number of the nearest neighbors, k , as: 10, 50, 100 and 250. The results are shown in Table 7, for a representative sub-collection of the datasets, all of which are computationally demanding. As can be seen, this approach provides by far the largest computational gains. In fact, for these datasets, each classifier required by the approximate IREOS has been computed on average with at most around 2% of the original runtime in the case of the largest neighborhood, $k = 250$. Except for dataset Wilt with $k = 10$ (the smallest neighborhood), the observed absolute difference (*i.e.*, the approximation error) Diff was in the second decimal digit.

Table 8 – Comparison between IREOS with and without fast, approximate strategies. The reported runtimes are the average times needed to compute the entire index assuming a sequential (non-parallelized) implementation.

Dataset	No Approximation	Combined Approximate Strategies	
	Runtime Index	Runtime Index	Diff
Anthyroid	2348:54:58	0:4:5 ($\approx 0\%$)	0.00265
Arrhythmia	0:13:56	0:1:45 (12.64%)	0.00356
Cardiotocography	18:26:45	0:1:24 (0.13%)	0.01707
Glass	0:2:9	0:0:18 (14%)	0.00182
HeartDisease	0:1:0	0:0:14 (23.53%)	0.00467
Hepatitis	0:0:7	0:0:1 (19.07%)	0.0033
InternetAds	1061:10:47	1:52:59 (0.18%)	0.00989
Ionosphere	0:13:47	0:1:4 (7.83%)	0.00924
Isolet	93:21:28	0:10:9 (0.18%)	0.0078
Lymphography	0:0:55	0:0:9 (16.92%)	0.00453
MultipleFeature	8:24:43	0:4:38 (0.92%)	0.00558
OpticalDigits	8:51:46	0:1:57 (0.37%)	0.00785
PageBlocks	491:47:43	0:2:24 (0.01%)	0.00172
Parkinson	0:0:3	0:0:0 (18.89%)	0.00222
Pima	0:30:43	0:0:45 (2.47%)	0.0079
Shuttle	3:18:20	0:0:39 (0.33%)	0.0161
SpamBase	124:9:0	0:5:45 (0.08%)	0.01186
Stamps	0:7:3	0:0:23 (5.53%)	0.00965
Vowel	0:0:40	0:0:1 (3.06%)	0.05453
Waveform	179:39:0	0:3:59 (0.04%)	0.00972
WBC	0:2:32	0:0:21 (13.88%)	0.00382
WDBC	0:13:19	0:0:42 (5.37%)	0.004
Wilt	359:23:27	0:2:8 (0.01%)	0.00333
WPBC	0:4:4	0:0:20 (8.57%)	0.00299

The results of an experiment that combines all the speedup strategies simultaneously are summarized in Table 8. In this case, for each dataset we report the average time to compute the entire index for one candidate solution, if a single processing core was used.² The estimated error for adaptive quadrature and the number of nearest neighbors were set to 0.005 and 250, respectively. As can be seen, with exception of the InternetAds dataset, IREOS can be computed in a few minutes without any major loss of accuracy, even considering a sequential (rather than parallelized) implementation.

Table 9 – Spearman correlation between approximate IREOS (speedup strategies combined) and ROC AUC.

Dataset	$m_{cl} = 1$	$m_{cl} = \sqrt{5\% \cdot N}$	$m_{cl} = n$
ALOI	-0.115	-0.115	0.103
Annthroid	0.248	-0.285	-0.37
Arrhythmia	0.539	0.673	0.721
Cardiotocography	0.758	0.915	0.976
Glass	0.467	0.479	0.394
HeartDisease	0.927	0.903	0.903
Hepatitis	0.939	0.903	0.903
InternetAds	0.709	0.891	0.721
Ionosphere	0.648	0.721	0.709
Isolet	0.2	0.588	0.588
KDDCup99	0.855	0.782	0.733
Lymphography	0.661	0.721	0.758
MultipleFeature	0.212	0.503	0.503
OpticalDigits	0.648	0.782	0.782
PageBlocks	0.927	0.964	0.952
Parkinson	0.515	0.515	0.515
PenDigits	0.6	0.661	0.661
Pima	0.758	0.879	0.879
Shuttle	0.867	0.867	0.867
SpamBase	0.794	0.842	0.842
Stamps	0.588	0.636	0.685
Vowel	0.927	0.709	0.648
Waveform	0.794	0.855	0.891
WBC	-0.018	0.042	0.042
WDBC	0.915	0.915	0.915
Wilt	-0.794	-0.903	-0.915
WPBC	0.479	0.648	0.806

At last, we have repeated the experiments of the previous chapter, now using all speedup approaches combined. The correlation experiment is shown in Table 9, whereas the model selection experiment is shown in Table 10 and Figure 33. When compared to the original results, where the index is computed without any approximation strategy (Tables 2, 3, and Figure 30), we can see that, apart from a few localized differences, the overall results remain stable and the conclusions do not change.

² Although our experiments have been run with our parallelized code in a multi-core server, we report the sum of the runtimes required to train classifiers across multiple processing nodes, as a worst-case scenario assuming a non-parallelized implementation.

Table 10 – Summarization of the ROC AUC values for all candidate solutions used for model selection. Three last columns indicate the solutions selected by approximate IREOS (all speedup strategies combined) for $m_{cl} = 1$, $m_{cl} = \sqrt{5\% \cdot N}$ and $m_{cl} = n$, respectively.

Dataset	Min	Max	Avg	IREOS ($m_{cl} = 1$)		IREOS ($m_{cl} = \sqrt{5\% \cdot N}$)		IREOS ($m_{cl} = n$)	
ALOI	0.544	0.805	0.675	0.632	KNN	0.632	KNN	0.632	KNN
Anthyroid	0.528	0.732	0.63	0.642	KNNW	0.574	KNN	0.596	LOF
Arrhythmia	0.5	0.916	0.737	0.879	GLOSH	0.879	GLOSH	0.879	GLOSH
Cardiotocography	0.536	0.839	0.689	0.74	LOF	0.74	LOF	0.839	LOF
Glass	0.492	0.904	0.699	0.539	GLOSH	0.539	GLOSH	0.539	GLOSH
HeartDisease	0.387	0.97	0.687	0.847	KNNW	0.847	KNNW	0.847	KNNW
Hepatitis	0.075	0.96	0.522	0.96	COF	0.96	COF	0.96	COF
InternetAds	0.413	0.895	0.656	0.843	FastABOD	0.843	FastABOD	0.843	FastABOD
Ionosphere	0.514	0.96	0.745	0.868	LoOP	0.868	LoOP	0.868	LoOP
Isolet	0.116	1	0.569	0.116	GLOSH	0.91	ODIN	0.91	ODIN
KDDCup99	0.364	0.99	0.68	0.923	KNNW	0.923	KNNW	0.923	KNNW
Lymphography	0.43	1	0.748	0.95	LDOF	0.95	LDOF	0.95	LDOF
MultipleFeature	0.233	0.994	0.638	0.918	LOF	0.918	LOF	0.918	LOF
OpticalDigits	0.164	0.999	0.596	0.999	KNN	0.737	LDF	0.737	LDF
PageBlocks	0.506	0.943	0.728	0.896	LOF	0.943	LDF	0.943	LDF
Parkinson	0.448	1	0.765	1	FastABOD	1	FastABOD	1	FastABOD
PenDigits	0.431	0.993	0.715	0.993	GLOSH	0.993	GLOSH	0.993	GLOSH
Pima	0.499	0.834	0.67	0.762	LOF	0.762	LOF	0.762	LOF
Shuttle	0.415	0.992	0.708	0.992	LDF	0.87	KNNW	0.87	KNNW
SpamBase	0.463	0.779	0.625	0.779	KNNW	0.779	KNNW	0.779	KNNW
Stamps	0.38	0.921	0.655	0.921	KNN	0.921	KNN	0.921	KNN
Vowel	0.223	1	0.614	1	FastABOD	1	FastABOD	0.916	COF
Waveform	0.5	0.796	0.66	0.708	INFLO	0.796	LDF	0.796	LDF
WBC	0.438	0.997	0.721	0.997	GLOSH	0.997	GLOSH	0.997	GLOSH
WDBC	0.497	0.988	0.745	0.988	GLOSH	0.988	GLOSH	0.988	GLOSH
Wilt	0.34	0.713	0.527	0.424	KNNW	0.424	KNNW	0.382	KNNW
WPBC	0.401	0.583	0.493	0.543	LDF	0.543	LDF	0.543	LDF

Figure 33 – Distribution of the ROC AUC values for all candidate solutions used in the experiments. The position of the solutions selected as best by approximate IREOS (all speedup strategies combined) are indicated by special symbols displayed in red.

6.5 Chapter Remarks

In this chapter, we proposed a number of speedup techniques to mitigate the computational burden, namely, numerical approximation of separability curves, pruning of outlier solution evaluations, and KNN guided sampling. These techniques allow close approximations of the exact index to be computed in a small fraction of the runtime. We designed three different experiments to assess the trade-off between computational efficiency and efficacy of the fast, approximate strategies to compute IREOS.

In the first experiment, we evaluated the approximation of the separability curves via the adaptive quadrature. Computing the index using adaptive quadrature can be seen as an improvement rather than an approximation of the index as it represents both a more efficient and a more effective computation. By using the adaptive quadrature, there is no need to set an arbitrary value for n_γ . Instead, the user has to specify a tolerance error, which is common practice in many machine learning algorithms that involve iterative optimization, such as ANNs, SVMs, KLR, k-means, among others. In the second experiment, we assessed the reduction of the computational cost when avoiding unnecessary computation via pruning. Pruning by $w_{(\cdot)}$ produces the exact index, and therefore, it should be performed whenever it is possible. In the third experiment, we evaluated the approximation of the index with the separability of each object computed using the object's k nearest neighbors only. This approximation provides by far the largest computational gains, reducing the classifier complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(k^3)$. Therefore, it should always be considered for very large datasets.

In addition to the speedup techniques provided in this chapter, it worth to remember that every single classifier can be trained simultaneously and in a completely independent way. In the case where we have $\mathcal{O}(n_c)$ computer cores for parallel processing, where n_c is the number of classifiers to be trained, the complexity of IREOS reduces to that of the classifier used. In other words, IREOS is highly parallelizable and can be implemented in a straightforward way in distributed (*e.g.*, cloud) environments using parallel computing frameworks such as MapReduce (DEAN; GHEMAWAT, 2008).

CONCLUSIONS

Outlier detection plays an important role in discovering patterns that are exceptional in some sense. Detecting such patterns is relevant for several reasons, *e.g.*, in many applications, these patterns represent extraordinary behaviors that deserve some special attention, such as employees with unusual productivity profiles, or customers with uncommon purchasing behaviors. Outlier detection techniques can be categorized in different ways, a common distinction is that between supervised, semi-supervised and unsupervised outlier detection techniques. In the semi-supervised scenario, the models are typically obtained using one-class classification techniques.

In this thesis, we focused on unsupervised outlier detection and one-class classification. Initially, we performed a comprehensive comparative study between algorithms of one-class classification and unsupervised outlier detection adapted to the semi-supervised setting. We provided recommendations of which algorithms to use in both general and specific application scenarios. We also discussed principles that should not be violated when adapting unsupervised outlier detection methods.

The main focus of this thesis, however, was on the unsupervised *evaluation* of unsupervised outlier detection and one-class classification, *i.e.*, automatically and without using external information (such as class labels), select the most appropriate models (algorithms, parameters), and evaluate the quality of the results. These problems have been investigated for decades in the area of data clustering, but are rarely mentioned in the area of outlier detection and one-class classification.

In the scenario of one-class classification, we proposed an (relative) internal evaluation measure based on the perturbation of the data. The measure is compared to others from the literature over a number of synthetic and real datasets in different experimental settings. Based on the experiments, we could recommend our measure as mostly suitable in general application scenarios, but we also recommended when it should not be used.

In the scenario of unsupervised outlier detection, we addressed the long-term open problem of internal and relative evaluation of outlier detection results. The pioneering measure for internal evaluation of binary outlier solutions, proposed by the author of this thesis in his master's work, was extended to the more general scenario of non-binary outlier solutions, which involves the evaluation of outlier detection scorings, which is the type of result produced by most widely used database-oriented algorithms in the literature.

Finally, due to the computational challenges resulting from the extension of the index, we provided strategies to avoid large amounts of unnecessary computations and mitigate the computational burden allowing close approximations of the exact index in a small fraction of the runtime.

7.1 Future Works and Limitations

In this thesis, we performed a comprehensive comparative study between algorithms of one-class classification and unsupervised outlier detection adapted to the semi-supervised setting. Based on the experiments performed in different experimental settings, we could recommend different algorithms for different application scenarios. However, no algorithm could outperform all others in every scenario. Therefore, it would be interesting to conduct a study of ensembles (ZIMEK; CAMPELLO; SANDER, 2013; TAX; DUIN, 2001a) for one-class classification combining methods of one-class classification and unsupervised outlier detection to increase the overall performance and robustness of the results.

In the scenario of one-class classification, we proposed an (relative) internal evaluation measure. Although we showed that our measure might be preferable over the others, this measure can be used only to compare the models in relative terms. It cannot provide a notion of quality in absolute terms, in order to assess whether the model is actually good or just not as bad as the others. An interesting research question to answer would be how to, for example, adjust the measure for chance, such as we did for IREOS. Notice, furthermore, that the problem of internal validation in absolute terms is still an open problem in the one-class classification literature, and the other measures from the literature presented in this thesis cannot provide such notion of quality either.

Regarding unsupervised outlier detection, we addressed the long-term open problem of internal and relative evaluation of outlier detection results. It is important to acknowledge that no single index can capture all possible facets of the unsupervised outlier detection problem. In fact, in the related area of data clustering, relying on multiple indexes is considered good practice and a variety of indexes have been proposed over the past few decades (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010). Some works in the clustering literature have taken advantage of such a diversity to build ensembles of validation indexes (JASKOWIAK *et al.*, 2016; VENDRAMIN;

[JASKOWIAK; CAMPELLO, 2013](#)). In the realm of unsupervised outlier detection, for which evaluation is still in its infancy, the development of new evaluation indexes is an important topic for future work. For instance, the use of different classifiers, other than those used in our paper, could give rise to new variants of IREOS. Different types of classifiers could lead to indexes with different biases, which nevertheless would still follow the same fundamental intuition behind IREOS. We hope, though, that our work will also stimulate the development of indexes supported on different grounds.

BIBLIOGRAPHY

AGGARWAL, C. C. **Outlier Analysis**. [S.l.]: Springer, 2013. Citation on page [24](#).

ANGIULLI, F.; FASSETTI, F. DOLPHIN: An efficient algorithm for mining distance-based outliers in very large datasets. **ACM Transactions on Knowledge Discovery from Data**, ACM, New York, USA, v. 3, n. 1, p. 4:1–4:57, 2009. Citation on page [76](#).

ANGIULLI, F.; PIZZUTI, C. Fast outlier detection in high dimensional spaces. In: **Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery**. Helsinki, Finland: [s.n.], 2002. (PKDD '02), p. 15–27. Citations on pages [33](#) and [99](#).

_____. Outlier mining in large high-dimensional data sets. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, New York, USA, v. 17, n. 2, p. 203–215, 2005. Citations on pages [33](#), [76](#), and [99](#).

ASCHER, U. M.; GREIF, C. **A First Course in Numerical Methods**. [S.l.]: Society for Industrial and Applied Mathematics, 2011. Citation on page [110](#).

BALDI, P.; BRUNAK, S.; CHAUVIN, Y.; ANDERSEN, C. A. F.; NIELSEN, H. Assessing the accuracy of prediction algorithms for classification: an overview. **Bioinformatics**, Oxford University Press, v. 16, n. 5, p. 412–424, 2000. Citations on pages [50](#) and [62](#).

BARNETT, V.; LEWIS, T. **Outliers in statistical data**. 3rd. ed. [S.l.]: John Wiley & Sons, 1994. Citations on pages [18](#), [25](#), [31](#), [32](#), [42](#), and [76](#).

BATISTA, G. E. A. P. A.; PRATI, R. C.; MONARD, M. C. A study of the behavior of several methods for balancing machine learning training data. **SIGKDD Explorations Newsletter**, ACM, New York, USA, v. 6, n. 1, p. 20–29, 2004. Citation on page [24](#).

BAY, S. D.; SCHWABACHER, M. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In: **Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. Washington, USA: [s.n.], 2003. (KDD '03), p. 29–38. Citation on page [76](#).

BAZARAA, M. S.; JARVIS, J. J.; SHERALI, H. D. **Linear Programming and Network Flows**. 4th. ed. [S.l.]: Wiley-Interscience, 2009. Citation on page [30](#).

BISHOP, C. M. **Pattern Recognition and Machine Learning**. [S.l.]: Springer, 2006. Citations on pages [23](#) and [26](#).

BONEAU, C. A. The effects of violations of assumptions underlying the t test. **Psychological Bulletin**, v. 57, n. 1, p. 49–64, 1960. Citation on page [94](#).

BREUNIG, M. M.; KRIEGEL, H.-P.; NG, R. T.; SANDER, J. LOF: Identifying density-based local outliers. In: **Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data**. New York, USA: [s.n.], 2000. (SIGMOD '00), p. 93–104. Citations on pages [35](#), [67](#), [76](#), [99](#), and [111](#).

BURDEN, R. L.; FAIRES, J. D.; BURDEN, A. M. **Numerical Analysis**. [S.l.]: Cengage learning, 2016. Citations on pages 108, 110, and 111.

CAMPELLO, R. J. G. B.; MOULAVI, D.; ZIMEK, A.; SANDER, J. Hierarchical density estimates for data clustering, visualization, and outlier detection. **ACM Transactions on Knowledge Discovery from Data**, ACM, New York, USA, v. 10, n. 1, p. 5:1–5:51, 2015. Citations on pages 38, 43, 59, 99, and 111.

CAMPOS, G. O.; ZIMEK, A.; SANDER, J.; CAMPELLO, R. J. G. B.; MICENKOVÁ, B.; SCHUBERT, E.; ASSENT, I.; HOULE, M. E. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. **Data Mining and Knowledge Discovery**, Springer, New York, USA, v. 30, n. 4, p. 891–927, 2016. Citations on pages 19, 24, 35, 40, 51, 62, 76, 78, 79, 96, 97, 98, 99, 102, and 105.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. **ACM Computing Surveys**, ACM, New York, USA, v. 41, n. 3, p. 15:1–15:58, 2009. Citation on page 17.

_____. Anomaly detection for discrete sequences: A survey. **IEEE Transactions on Knowledge & Data Engineering**, IEEE Educational Activities Department, Piscataway, USA, v. 24, n. 5, p. 823–839, 2012. Citation on page 17.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. SMOTE: Synthetic minority over-sampling technique. **Journal of Artificial Intelligence Research**, AI Access Foundation, USA, v. 16, n. 1, p. 321–357, 2002. Citation on page 24.

CRASWELL, N. Precision at n. In: _____. **Encyclopedia of Database Systems**. [S.l.]: Springer, 2009. p. 2127–2128. Citation on page 79.

DAVIS, P. J.; RABINOWITZ, P. **Methods of Numerical Integration**. 2nd. ed. [S.l.]: Academic Press, 1984. Citation on page 110.

DEAN, J.; GHEMAWAT, S. Mapreduce: Simplified data processing on large clusters. **Communications of the ACM**, ACM, New York, USA, v. 51, n. 1, p. 107–113, 2008. Citation on page 120.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. **Journal of Machine Learning Research**, JMLR.org, v. 7, p. 1–30, 2006. Citations on pages 52, 54, and 68.

DÉSIR, C.; BERNARD, S.; PETITJEAN, C.; HEUTTE, L. One class random forests. **Pattern Recognition**, Elsevier, v. 46, n. 12, p. 3490 – 3506, 2013. Citation on page 50.

DHEERU, D.; TANISKIDOU, E. K. **UCI Machine Learning Repository**. 2017. Available: <<http://archive.ics.uci.edu/ml>>. Citations on pages 48 and 49.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern Classification**. 2nd. ed. [S.l.]: Wiley-Interscience, 2000. Citations on pages 23 and 61.

DUIN, R. P. W. On the choice of smoothing parameters for parzen estimators of probability density functions. **IEEE Transactions on Computers**, C-25, n. 11, p. 1175–1179, 1976. Citations on pages 27 and 62.

ERFANI, S. M.; BAKTASHMOTLAGH, M.; RAJASEGARAR, S.; KARUNASEKERA, S.; LECKIE, C. R1SVM: a randomised nonlinear approach to large-scale anomaly detection. In: **Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2015. (AAAI'15), p. 432–438. Citation on page [67](#).

FAWCETT, T. Roc graphs: Notes and practical considerations for researchers. **Machine learning**, v. 31, n. 1, p. 1–38, 2004. Citation on page [51](#).

FAYYAD, U. M.; SHAPIRO, G. P.; SMYTH, P.; UTHURUSAMY, R. **Advances in Knowledge Discovery and Data Mining**. [S.l.]: American Association for Artificial Intelligence, 1996. Citation on page [17](#).

FREEMAN, J. A.; SKAPURA, D. M. **Neural Networks: Algorithms, Applications, and Programming Techniques**. [S.l.]: Addison Wesley Longman Publishing, 1991. Citation on page [31](#).

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. **Journal of the American Statistical Association**, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937. Citation on page [52](#).

GHAFOORI, Z.; RAJASEGARAR, S.; ERFANI, S. M.; KARUNASEKERA, S.; LECKIE, C. A. Unsupervised parameter estimation for one-class support vector machines. In: **Proceedings of the Advances in Knowledge Discovery and Data Mining**. Auckland, New Zealand: [s.n.], 2016. (PAKDD '16), p. 183–195. Citation on page [62](#).

GHOTING, A.; PARTHASARATHY, S.; OTEY, M. E. Fast mining of distance-based outliers in high-dimensional datasets. **Data Mining and Knowledge Discovery**, Kluwer Academic Publishers, Hingham, MA, USA, v. 16, n. 3, p. 349–364, 2008. Citation on page [76](#).

GIONIS, A.; INDYK, P.; MOTWANI, R. Similarity search in high dimensions via hashing. In: **Proceedings of the 25th International Conference on Very Large Data Bases**. Edinburgh, UK: [s.n.], 1999. (VLDB '99), p. 518–529. Citation on page [114](#).

GONNET, P. A review of error estimation in adaptive quadrature. **ACM Computing Surveys**, ACM, New York, USA, v. 44, n. 4, p. 22:1–22:36, 2012. Citations on pages [108](#) and [110](#).

GRUBBS, F. E. Procedures for detecting outlying observations in samples. **Technometrics**, Taylor & Francis, v. 11, n. 1, p. 1–21, 1969. Citation on page [18](#).

HADI, A. S. Identifying multiple outliers in multivariate data. **Journal of the Royal Statistical Society. Series B (Methodological)**, Wiley for the Royal Statistical Society, v. 54, n. 3, p. 761–771, 1992. Citation on page [33](#).

HADI, A. S.; IMON, A. H. M. R.; WERNER, M. Detection of outliers. **Wiley Interdisciplinary Reviews: Computational Statistics**, John Wiley & Sons, v. 1, n. 1, p. 57–70, 2009. Citation on page [17](#).

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. On clustering validation techniques. **Journal of Intelligent Information Systems**, Kluwer Academic Publishers, Hingham, USA, v. 17, n. 2-3, p. 107–145, 2001. Citations on pages [19](#) and [77](#).

HAN, J.; KAMBER, M. **Data Mining: Concepts and Techniques**. 2nd. ed. [S.l.]: Morgan Kaufmann, 2006. Citations on pages [17](#) and [18](#).

HAN, J.; KAMBER, M.; PEI, J. **Data Mining: Concepts and Techniques**. 3rd. ed. [S.l.]: Morgan Kaufmann, 2011. Citations on pages 63 and 76.

HANDL, J.; KNOWLES, J.; KELL, D. B. Computational cluster validation in post-genomic data analysis. **Bioinformatics**, Oxford University Press, v. 21, n. 15, p. 3201–3212, 2005. Citations on pages 48 and 49.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2nd. ed. [S.l.]: Springer-Verlag New York, 2013. Citations on pages 80, 86, and 89.

HAUTAMÄKI, V.; KÄRKKÄINEN, I.; FRÄNTI, P. Outlier detection using k-nearest neighbour graph. In: **Proceedings of the 17th International Conference on Pattern Recognition**. Washington, USA: [s.n.], 2004. (ICPR '04), p. 430–433. Citations on pages 34 and 99.

HAWKINS, D. M. **Identification of outliers**. [S.l.]: Chapman and Hall, 1980. Citations on pages 18, 25, 32, 76, and 77.

HODGE, V.; AUSTIN, J. A survey of outlier detection methodologies. **Artificial Intelligence Review**, Kluwer Academic Publishers, Norwell, USA, v. 22, n. 2, p. 85–126, 2004. Citation on page 17.

JAIN, A. K.; DUBES, R. C. **Algorithms for Clustering Data**. [S.l.]: Prentice-Hall, 1988. Citations on pages 19, 77, and 95.

JANSSENS, J. H. M.; FLESCHE, I.; POSTMA, E. O. Outlier detection with one-class classifiers from ML and KDD. In: **Proceedings of the 2009 International Conference on Machine Learning and Applications**. Miami Beach, USA: [s.n.], 2009. (ICMLA '09), p. 147–153. Citations on pages 41, 42, 49, 53, 54, 58, and 59.

JANSSENS, J. H. M.; POSTMA, E. O. One-class classification with LOF and LOCI: An empirical comparison. In: **Proceedings of the 18th Annual Belgian-Dutch on Machine Learning**. Tilburg, The Netherlands: [s.n.], 2009. p. 56–64. Citation on page 42.

JAPKOWICZ, N.; MYERS, C.; GLUCK, M. A novelty detection approach to classification. In: **Proceedings of the 14th International Joint Conference on Artificial Intelligence**. Montreal, Canada: [s.n.], 1995. (IJCAI'95), p. 518–523. Citation on page 31.

JASKOWIAK, P. A.; MOULAVI, D.; FURTADO, A. C. S.; CAMPELLO, R. J. G. B.; ZIMEK, A.; SANDER, J. On strategies for building effective ensembles of relative clustering validity criteria. **Knowledge and Information Systems**, v. 47, n. 2, p. 329–354, 2016. Citations on pages 122 and 123.

JIN, W.; TUNG, A. K. H.; HAN, J. Mining top-n local outliers in large databases. In: **Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. San Francisco, California: [s.n.], 2001. (KDD '01), p. 293–298. Citation on page 76.

JIN, W.; TUNG, A. K. H.; HAN, J.; WANG, W. Ranking outliers using symmetric neighborhood relationship. In: **Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining**. Singapore: [s.n.], 2006. (PAKDD' 06), p. 577–593. Citations on pages 36, 76, and 99.

JOACHIMS, T. Training linear SVMs in linear time. In: **Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. Philadelphia, USA: [s.n.], 2006. (KDD '06), p. 217–226. Citation on page [67](#).

JUSZCZAK, P. **Learning to recognise: A study on one-class classification and active learning**. Phd Thesis (PhD Thesis) — Delft University of Technology, 2006. Citation on page [64](#).

JUSZCZAK, P.; TAX, D. M.; PEKALSKA, E.; DUIN, R. P. Minimum spanning tree based one-class classifier. **Neurocomputing**, Elsevier, v. 72, n. 7, p. 1859 – 1869, 2009. Citation on page [68](#).

KEERTHI, S. S.; DUAN, K. B.; SHEVADE, S. K.; POO, A. N. A fast dual algorithm for kernel logistic regression. **Machine Learning**, Kluwer Academic Publishers, Hingham, USA, v. 61, n. 1-3, p. 151–165, 2005. Citation on page [113](#).

Pearson's correlation coefficient. In: KIRCH, W. (Ed.). **Encyclopedia of Public Health**. Dordrecht: Springer Netherlands, 2008. p. 1090–1091. Citation on page [50](#).

KNORR, E. M.; NG, R. T. Algorithms for mining distance-based outliers in large datasets. In: **Proceedings of the 24rd International Conference on Very Large Data Bases**. New York City, USA: [s.n.], 1998. (VLDB '98), p. 392–403. Citations on pages [18](#), [32](#), and [76](#).

KNORR, E. M.; NG, R. T.; TUCAKOV, V. Distance-based outliers: Algorithms and applications. **The VLDB Journal**, Springer-Verlag New York, Secaucus, USA, v. 8, n. 3-4, p. 237–253, 2000. Citations on pages [33](#) and [76](#).

KOLLIOS, G.; GUNOPULOS, D.; KOUDAS, N.; BERCHTOLD, S. Efficient biased sampling for approximate clustering and outlier detection in large data sets. **IEEE Transactions on Knowledge & Data Engineering**, IEEE Educational Activities Department, Piscataway, USA, v. 15, n. 5, p. 1170–1187, 2003. Citation on page [76](#).

KRIEGEL, H.-P.; KRÖGER, P.; SCHUBERT, E.; ZIMEK, A. LoOP: Local outlier probabilities. In: **Proceedings of the 18th ACM Conference on Information and Knowledge Management**. Hong Kong, China: [s.n.], 2009. (CIKM '09), p. 1649–1652. Citations on pages [37](#), [76](#), and [99](#).

_____. Interpreting and unifying outlier scores. In: **Proceedings of the 2011 SIAM International Conference on Data Mining**. Arizona, USA: [s.n.], 2011. (SDM '11), p. 13–24. Citations on pages [76](#), [83](#), [84](#), [85](#), [100](#), and [111](#).

KRIEGEL, H.-P.; SCHUBERT, M.; ZIMEK, A. Angle-based outlier detection in high-dimensional data. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. Las Vegas, USA: [s.n.], 2008. (KDD '08), p. 444–452. Citations on pages [39](#), [76](#), and [99](#).

KUBAT, M.; MATWIN, S. Addressing the curse of imbalanced training sets: One-sided selection. In: **Proceedings of the 14th International Conference on Machine Learning**. [S.l.: s.n.], 1997. p. 179–186. Citation on page [24](#).

KUNCIR, G. F. Algorithm 103: Simpson's rule integrator. **Communications of the ACM**, ACM, New York, USA, v. 5, n. 6, p. 347, 1962. Citation on page [108](#).

LATECKI, L. J.; LAZAREVIC, A.; POKRAJAC, D. Outlier detection with kernel density functions. In: **Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition**. Leipzig, Germany: [s.n.], 2007. (MLDM '07), p. 61–75. Citations on pages 37 and 99.

LEE, Y.-J.; MANGASARIAN, O. L. RSVM: Reduced support vector machines. In: **Proceedings of the 2001 SIAM International Conference on Data Mining**. Chicago, USA: [s.n.], 2001. (SDM '01), p. 1–17. Citation on page 113.

MAHALANOBIS, P. C. On the generalized distance in statistics. In: **Proceedings of the National Institute of Sciences of India**. India: [s.n.], 1936. p. 49–55. Citation on page 26.

MARKOU, M.; SINGH, S. Novelty detection: a review—part 1: statistical approaches. **Signal Processing**, v. 83, n. 12, p. 2481–2497, 2003. Citations on pages 18 and 42.

MARQUES, H. O. **Avaliação e seleção de modelos em detecção não supervisionada de outliers**. Master's Thesis (Master's Thesis) — Universidade de São Paulo, 2015. Citations on pages 19, 20, 78, 79, 105, and 107.

MARQUES, H. O.; CAMPELLO, R. J. G. B.; ZIMEK, A.; SANDER, J. On the internal evaluation of unsupervised outlier detection. In: **Proceedings of the 27th International Conference on Scientific and Statistical Database Management**. San Diego, USA: [s.n.], 2015. (SSDBM '15), p. 7:1–7:12. Citations on pages 19, 20, 48, 78, 79, 98, 101, and 107.

MATTHEWS, B. W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. **Biochimica et Biophysica Acta (BBA) - Protein Structure**, Elsevier, v. 405, n. 2, p. 442 – 451, 1975. Citation on page 50.

MICENKOVÁ, B.; NG, R. T.; DANG, X. H.; ASSENT, I. Explaining outliers by subspace separability. In: **Proceedings of the 13th International Conference on Data Mining**. Dallas, USA: [s.n.], 2013. (ICDM '13), p. 518–527. Citation on page 114.

MILLIGAN, G. W.; COOPER, M. C. An examination of procedures for determining the number of clusters in a data set. **Psychometrika**, v. 50, n. 2, p. 159–179, 1985. Citations on pages 19 and 77.

MOORE, G. The future of integrated electronics. **Fairchild Semiconductor internal publication**, 1964. Citation on page 17.

_____. Cramming more components onto integrated circuits. **Electronics Magazine**, Morgan Kaufmann Publishers, San Francisco, USA, v. 38, n. 8, p. 114–117, 1965. Citation on page 17.

_____. Progress in digital integrated electronics. **International Electron Devices Meeting**, v. 21, p. 11–13, 1975. Citation on page 17.

NEMÉNYI, P. **Distribution-free multiple comparisons**. Phd Thesis (PhD Thesis) — Princeton University, 1963. Citation on page 53.

ORAIR, G. H.; TEIXEIRA, C. H. C.; MEIRA JR., W.; WANG, Y.; PARTHASARATHY, S. Distance-based outlier detection: Consolidation and renewed bearing. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 3, n. 1-2, p. 1469–1480, 2010. Citation on page 76.

PANDA, N.; CHANG, E. Y.; WU, G. Concept boundary detection for speeding up SVMs. In: **Proceedings of the 23rd International Conference on Machine Learning**. Pittsburgh, USA: [s.n.], 2006. (ICML '06), p. 681–688. Citations on pages [113](#) and [114](#).

PANG, G.; CAO, L.; CHEN, L.; LIU, H. Learning homophily couplings from non-iid data for joint feature selection and noise-resilient outlier detection. In: **Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence**. Melbourne, Australia: [s.n.], 2017. (IJCAI'17), p. 2585–2591. Citation on page [105](#).

PAPADIMITRIOU, S.; KITAGAWA, H.; GIBBONS, P. B.; FALOUTSOS, C. LOCI: fast outlier detection using the local correlation integral. In: **Proceedings of the 19th International Conference on Data Engineering**. Bangalore, India: [s.n.], 2003. (ICDE '03), p. 315–326. Citations on pages [38](#), [76](#), and [111](#).

PARZEN, E. On estimation of a probability density function and mode. **The annals of mathematical statistics**, JSTOR, v. 33, n. 3, p. 1065–1076, 1962. Citation on page [26](#).

PATCHA, A.; PARK, J.-M. An overview of anomaly detection techniques: Existing solutions and latest technological trends. **Computer Networks**, Elsevier North-Holland, New York, USA, v. 51, n. 12, p. 3448–3470, 2007. Citation on page [17](#).

PEKALSKA, E.; TAX, D. M. J.; DUIN, R. P. W. One-class LP classifier for dissimilarity representations. In: **Proceedings of the 15th International Conference on Neural Information Processing Systems**. Canada: [s.n.], 2002. (NIPS'02), p. 777–784. Citation on page [29](#).

PIMENTEL, M. A. F.; CLIFTON, D. A.; CLIFTON, L.; TARASSENKO, L. Review: A review of novelty detection. **Signal Process.**, Elsevier North-Holland, Amsterdam, The Netherlands, v. 99, p. 215–249, 2014. Citations on pages [18](#) and [25](#).

PLATT, J. **Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines**. [S.l.], 1998. Citation on page [113](#).

RADOVANOVIĆ, M.; NANOPOULOS, A.; IVANOVIĆ, M. Reverse nearest neighbors in unsupervised distance-based outlier detection. **IEEE Transactions on Knowledge & Data Engineering**, IEEE Computer Society, Los Alamitos, USA, v. 27, n. 05, p. 1369–1382, 2015. Citation on page [34](#).

RAMASWAMY, S.; RASTOGI, R.; SHIM, K. Efficient algorithms for mining outliers from large data sets. In: **Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data**. New York, USA: [s.n.], 2000. (SIGMOD '00), p. 427–438. Citations on pages [33](#), [67](#), [76](#), and [99](#).

RAND, W. M. Objective criteria for the evaluation of clustering methods. **Journal of the American Statistical Association**, Taylor & Francis, v. 66, n. 336, p. 846–850, 1971. Citations on pages [64](#) and [65](#).

ROSS, S. M. **Introduction to Probability Models**. 10th. ed. [S.l.]: Elsevier, 2009. Citation on page [94](#).

ROUSSEEUW, P. J.; ZOMEREN, B. C. v. Unmasking multivariate outliers and leverage points. **Journal of the American Statistical Association**, American Statistical Association, v. 85, n. 411, p. 633–639, 1990. Citation on page [33](#).

SCHÖLKOPF, B.; PLATT, J. C.; SHAWE-TAYLOR, J. C.; SMOLA, A. J.; WILLIAMSON, R. C. Estimating the support of a high-dimensional distribution. **Neural Computation**, MIT Press, Cambridge, USA, v. 13, n. 7, p. 1443–1471, 2001. Citations on pages 27 and 28.

SCHÖLKOPF, B.; SMOLA, A. J. **Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**. [S.l.]: MIT Press, 2001. Citations on pages 80 and 86.

SCHUBERT, E.; WOJDANOWSKI, R.; ZIMEK, A.; KRIEGEL, H.-P. On evaluation of outlier rankings and outlier scores. In: **Proceedings of the 2012 SIAM International Conference on Data Mining**. California, USA: [s.n.], 2012. (SDM '12), p. 1047–1058. Citations on pages 19, 24, 79, and 83.

SCHUBERT, E.; ZIMEK, A.; KRIEGEL, H.-P. Generalized outlier detection with flexible kernel density estimates. In: **Proceedings of the 2014 SIAM International Conference on Data Mining**. Philadelphia, USA: [s.n.], 2014. (SDM '14), p. 542–550. Citations on pages 37 and 99.

_____. Local outlier detection reconsidered: A generalized view on locality with applications to spatial, video, and network outlier detection. **Data Mining and Knowledge Discovery**, Springer, New York, USA, v. 28, n. 1, p. 190–237, 2014. Citations on pages 35, 76, and 99.

_____. Fast and scalable outlier detection with approximate nearest neighbor ensembles. In: **Proceedings of the 20th International Conference on Database Systems for Advanced Applications**. Hanoi, Vietnam: [s.n.], 2015. (DASFAA '15), p. 19–36. Citation on page 114.

STRANG, G. **Introduction to Linear Algebra**. 5th. ed. [S.l.]: Wellesley-Cambridge Press, 2016. Citation on page 26.

SWERSKY, L. **A Study of Unsupervised Outlier Detection for One-Class Classification**. Master's Thesis (Master's Thesis) — University of Alberta, 2018. Citations on pages 41 and 47.

SWERSKY, L.; MARQUES, H. O.; SANDER, J.; CAMPELLO, R. J. G. B.; ZIMEK, A. On the evaluation of outlier detection and one-class classification methods. In: **Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics**. Montreal, Canada: [s.n.], 2016. (DSAA '16), p. 1–10. Citation on page 41.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. 1st. ed. [S.l.]: Addison Wesley, 2005. Citations on pages 17, 18, 33, 58, 61, and 63.

TANG, J.; CHEN, Z.; FU, A. W.-c.; CHEUNG, D. W. Enhancing effectiveness of outlier detections for low density patterns. In: **Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining**. Taipei, Taiwan: [s.n.], 2002. (PAKDD' 02), p. 535–548. Citations on pages 36, 76, and 99.

TANG, Y.; ZHANG, Y.-Q.; CHAWLA, N. V.; KRASSER, S. SVMs modeling for highly imbalanced classification. **Transactions on Systems, Man, and Cybernetics Part B**, IEEE Press, v. 39, n. 1, p. 281–288, 2009. Citation on page 24.

TAX, D. M. **DDtools, the Data Description Toolbox for Matlab**. 2018. Version 2.1.3. Citations on pages 48 and 68.

TAX, D. M. J. **One-class classification**. Phd Thesis (PhD Thesis) — Delft University of Technology, 2001. Citations on pages 25, 28, 29, 30, 40, and 59.

TAX, D. M. J.; DUIN, R. P. W. Combining one-class classifiers. In: **International Workshop on Multiple Classifier Systems**. [S.l.: s.n.], 2001. (MCS '01), p. 299–308. Citation on page [122](#).

_____. Uniform object generation for optimizing one-class classifiers. **The Journal of Machine Learning Research**, JMLR.org, v. 2, p. 155–173, 2001. Citations on pages [62](#), [64](#), [67](#), [68](#), and [73](#).

_____. Support vector data description. **Machine Learning**, Kluwer Academic Publishers, Hingham, USA, v. 54, n. 1, p. 45–66, 2004. Citations on pages [28](#), [64](#), [66](#), and [67](#).

TAX, D. M. J.; MÜLLER, K.-R. A consistency-based model selection for one-class classification. In: **Proceedings of the 17th International Conference on Pattern Recognition**. Cambridge, UK: [s.n.], 2004. (ICPR '04), p. 363–366 Vol.3. Citation on page [62](#).

TING, K. M. An instance-weighting method to induce cost-sensitive trees. **IEEE Transactions on Knowledge & Data Engineering**, IEEE Educational Activities Department, Piscataway, USA, v. 14, n. 3, p. 659–665, 2002. ISSN 1041-4347. Citation on page [24](#).

TRIOLA, M. F. **Elementary Statistics**. 10th. ed. [S.l.]: Pearson, 2007. Citations on pages [93](#), [94](#), and [95](#).

TRITTENBACH, H.; ENGLHARDT, A.; BÖHM, K. An overview and a benchmark of active learning for one-class classification. **arXiv preprint 1808.04759**, 2018. Citation on page [68](#).

VAPNIK, V. N. **The Nature of Statistical Learning Theory**. [S.l.]: Springer-Verlag New York, 1995. Citation on page [27](#).

VENDRAMIN, L.; CAMPELLO, R. J. G. B.; HRUSCHKA, E. R. Relative clustering validity criteria: A comparative overview. **Statistical Analysis and Data Mining**, John Wiley & Sons, New York, USA, v. 3, n. 4, p. 209–235, 2010. Citations on pages [19](#), [77](#), and [122](#).

VENDRAMIN, L.; JASKOWIAK, P. A.; CAMPELLO, R. J. G. B. On the combination of relative clustering validity criteria. In: **Proceedings of the 25th International Conference on Scientific and Statistical Database Management**. Baltimore, USA: [s.n.], 2013. (SSDBM '13), p. 4:1–4:12. Citations on pages [19](#), [77](#), [122](#), and [123](#).

VU, N. H.; GOPALKRISHNAN, V. Efficient pruning schemes for distance-based outlier detection. In: **Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases**. Bled, Slovenia: [s.n.], 2009. (ECMLPKDD '09), p. 160–175. Citation on page [76](#).

WANG, S.; LIU, Q.; ZHU, E.; PORIKLI, F.; YIN, J. Hyperparameter selection of one-class support vector machine by self-adaptive data shifting. **Pattern Recogn.**, Elsevier Science Inc., New York, USA, v. 74, n. C, p. 198–211, 2018. Citations on pages [50](#), [62](#), [64](#), [67](#), [68](#), and [73](#).

WEBB, G. I. Lazy learning. In: SAMMUT, C.; WEBB, G. I. (Ed.). **Encyclopedia of Machine Learning**. Boston, MA: Springer, 2011. p. 571–572. Citation on page [67](#).

WITTEN, I. H.; FRANK, E.; HALL, M. A. **Data Mining: Practical Machine Learning Tools and Techniques**. Third edition. [S.l.]: Morgan Kaufmann, 2011. Citation on page [17](#).

XIAO, Y.; WANG, H.; XU, W. Parameter selection of gaussian kernel for one-class svm. **IEEE Transactions on Cybernetics**, v. 45, n. 5, p. 941–953, 2015. Citation on page [62](#).

ZHANG, K.; HUTTER, M.; JIN, H. A new local distance-based outlier detection approach for scattered real-world data. In: **Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining**. Bangkok, Thailand: [s.n.], 2009. (PAKDD '09), p. 813–822. Citations on pages [34](#), [76](#), [99](#), and [111](#).

ZHU, J.; HASTIE, T. Kernel logistic regression and the import vector machine. **Journal of Computational and Graphical Statistics**, Taylor & Francis, v. 14, n. 1, p. 185–205, 2005. Citations on pages [80](#), [86](#), and [89](#).

ZIMEK, A.; CAMPELLO, R. J.; SANDER, J. Ensembles for unsupervised outlier detection: Challenges and research questions a position paper. **SIGKDD Explorations Newsletter**, ACM, New York, USA, v. 15, n. 1, p. 11–22, 2013. Citations on pages [19](#), [77](#), [79](#), [105](#), and [122](#).

ZIMEK, A.; CAMPELLO, R. J. G. B.; SANDER, J. Data perturbation for outlier detection ensembles. In: **Proceedings of the 26th International Conference on Scientific and Statistical Database Management**. Aalborg, Denmark: [s.n.], 2014. (SSDBM '14), p. 13:1–13:12. Citations on pages [48](#), [49](#), [57](#), [64](#), [65](#), and [72](#).

ZIMEK, A.; GAUDET, M.; CAMPELLO, R. J.; SANDER, J. Subsampling for efficient and effective unsupervised outlier detection ensembles. In: **Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, USA: [s.n.], 2013. (KDD '13), p. 428–436. Citations on pages [19](#), [48](#), [49](#), [57](#), and [72](#).

ZIMEK, A.; SCHUBERT, E.; KRIEGEL, H.-P. A survey on unsupervised outlier detection in high-dimensional numerical data. **Statistical Analysis and Data Mining**, John Wiley & Sons, New York, USA, v. 5, n. 5, p. 363–387, 2012. Citations on pages [17](#) and [19](#).

TABLES OF RESULTS, CHAPTER 3

The tables below show the detailed values for the results discussed in chapter 3. The tables display ROC AUC and MCC values for each method in the experiments performed in chapter 3. The highest achieved values for each data set are shown in bold.

Table 11 – First type of experiments — Unimodal — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-4c-no1	0.999	0.997	1	0.999	0.999	0.997	0.998	0.999	0.997	0.999	0.999
2d-4c-no2	0.996	0.992	0.997	0.997	0.997	0.994	0.994	0.998	0.998	0.997	0.997
2d-4c-no3	1	0.998	1	1	1	0.998	0.996	0.999	0.999	1	1
2d-4c-no4	1	0.999	1	1	1	1	0.996	1	1	1	1
2d-4c-no5	0.999	0.996	0.999	0.999	0.999	0.997	0.998	0.999	0.992	0.999	0.999
2d-4c-no6	0.999	0.999	1	0.999	1	0.998	0.994	0.999	0.997	0.999	1
2d-4c-no7	0.997	0.997	1	0.998	0.999	0.998	0.996	0.999	0.995	0.999	0.999
2d-4c-no8	0.999	0.980	1	1	1	0.996	1	1	1	1	0.998
2d-4c-no9	0.996	0.996	0.997	0.998	0.998	0.995	0.994	0.998	0.997	0.997	0.999
2d-10c-no0	0.999	0.996	0.999	0.999	0.999	0.997	0.999	0.999	0.999	0.999	0.999
2d-10c-no1	0.999	0.999	1	1	1	0.999	0.998	1	1	1	1
2d-10c-no2	1	0.999	1	1	1	0.999	0.999	1	0.999	1	1
2d-10c-no3	0.999	0.998	1	1	1	0.999	0.998	1	0.999	0.999	1
2d-10c-no4	0.998	0.997	1	0.999	0.999	0.997	0.996	0.999	0.999	0.999	0.999
2d-10c-no5	0.987	0.983	0.991	0.989	0.991	0.983	0.988	0.990	0.984	0.990	0.989
2d-10c-no6	0.999	0.996	0.999	0.999	0.999	0.996	0.995	0.999	0.996	0.999	0.999
2d-10c-no7	0.998	0.997	1	0.999	0.999	0.997	0.998	0.999	0.999	0.998	0.999
2d-10c-no8	0.994	0.985	0.996	0.995	0.995	0.989	0.991	0.994	0.991	0.996	0.997
2d-10c-no9	0.999	0.992	0.999	0.999	0.999	0.995	0.998	0.999	0.998	0.999	0.999
2d-20c-no0	1	0.997	1	1	1	0.999	1	1	1	1	1
2d-20c-no1	1	1	1	1	1	0.999	1	0.999	1	1	1
2d-20c-no2	0.999	0.996	0.998	0.999	0.999	0.998	1	0.999	1	0.999	0.999
2d-20c-no3	1	0.999	1	1	1	0.999	0.999	1	1	1	1
2d-20c-no4	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-20c-no5	0.999	0.996	0.999	0.999	0.999	0.994	0.998	0.998	0.999	0.999	0.999
2d-20c-no6	0.999	0.998	0.999	0.999	0.999	0.998	0.999	0.999	1	0.999	0.999
2d-20c-no7	1	0.997	0.999	0.999	1	0.998	1	0.999	0.999	1	0.999
2d-20c-no8	1	0.997	0.999	0.999	1	0.998	0.998	0.999	0.998	0.999	1
2d-20c-no9	1	0.997	0.998	1	1	0.998	1	0.998	1	1	1
2d-40c-no0	1	0.998	1	1	1	0.999	0.999	0.999	1	1	1
2d-40c-no1	0.999	0.998	0.999	0.999	0.999	0.999	0.999	0.999	1	1	0.999
2d-40c-no2	0.999	0.997	0.999	0.999	0.999	0.998	0.999	0.999	0.999	0.999	0.999
2d-40c-no3	0.999	0.998	0.999	0.999	0.999	0.997	0.999	0.998	0.996	0.999	0.999
2d-40c-no4	1	0.998	0.999	1	1	0.999	0.999	0.999	1	1	1
2d-40c-no5	1	0.998	0.999	0.999	1	0.998	0.999	0.999	1	1	1
2d-40c-no6	1	0.998	0.999	1	1	0.998	0.999	0.999	1	1	1
2d-40c-no7	1	0.997	0.999	0.999	1	0.998	0.999	0.999	1	1	1
2d-40c-no8	0.999	0.997	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
2d-40c-no9	1	0.998	0.999	1	1	0.999	0.999	0.999	1	1	1
	0.999	0.996	0.999	0.999	0.999	0.997	0.998	0.999	0.998	0.999	0.999

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	0.988	0.94	0.991	0.988	0.99	0.971	0.897	0.983	0.772	0.727	0.983
2d-4c-no1	0.915	0.953	0.991	0.956	0.891	0.884	0.712	0.932	0.683	0.633	0.964
2d-4c-no2	0.875	0.88	0.926	0.898	0.887	0.855	0.697	0.878	0.731	0.646	0.931
2d-4c-no3	0.988	0.954	0.983	0.985	0.986	0.932	0.827	0.988	0.699	0.545	0.98
2d-4c-no4	0.996	0.876	0.991	0.991	0.994	0.964	0.75	0.986	0.46	0.614	0.97
2d-4c-no5	0.966	0.9	0.976	0.965	0.966	0.929	0.704	0.971	0.685	0.832	0.937
2d-4c-no6	0.987	0.934	0.984	0.985	0.986	0.971	0.812	0.985	0.701	0.648	0.986
2d-4c-no7	0.847	0.94	0.971	0.907	0.872	0.868	0.738	0.952	0.649	0.539	0.962
2d-4c-no8	0.874	0.853	0.961	0.933	0.937	0.839	0.746	0.957	0.57	0.604	0.966
2d-4c-no9	0.932	0.899	0.934	0.932	0.932	0.906	0.73	0.93	0.701	0.597	0.964
2d-10c-no0	0.978	0.937	0.965	0.976	0.977	0.953	0.869	0.967	0.822	0.644	0.962
2d-10c-no1	0.945	0.935	0.986	0.977	0.984	0.923	0.593	0.978	0.709	0.768	0.96
2d-10c-no2	0.988	0.933	0.992	0.99	0.993	0.933	0.79	0.983	0.779	0.598	0.978
2d-10c-no3	0.927	0.902	0.956	0.944	0.942	0.889	0.672	0.939	0.818	0.643	0.926
2d-10c-no4	0.876	0.928	0.969	0.919	0.909	0.902	0.761	0.926	0.868	0.612	0.969
2d-10c-no5	0.879	0.876	0.904	0.894	0.898	0.856	0.668	0.896	0.803	0.595	0.894
2d-10c-no6	0.87	0.904	0.922	0.881	0.884	0.843	0.589	0.88	0.805	0.633	0.942
2d-10c-no7	0.891	0.913	0.959	0.938	0.941	0.894	0.553	0.953	0.86	0.709	0.945
2d-10c-no8	0.856	0.839	0.864	0.868	0.871	0.822	0.788	0.864	0.818	0.74	0.913
2d-10c-no9	0.931	0.93	0.931	0.928	0.93	0.912	0.819	0.922	0.842	0.531	0.975
2d-20c-no0	0.967	0.947	0.981	0.96	0.965	0.908	0.902	0.942	0.698	0.772	0.954
2d-20c-no1	0.973	0.901	0.965	0.965	0.971	0.897	0.913	0.921	0.634	0.819	0.954
2d-20c-no2	0.912	0.841	0.915	0.927	0.93	0.859	0.828	0.893	0.61	0.813	0.909
2d-20c-no3	0.965	0.893	0.955	0.961	0.962	0.893	0.871	0.94	0.657	0.815	0.943
2d-20c-no4	0.93	0.89	0.947	0.946	0.95	0.894	0.82	0.931	0.7	0.803	0.94
2d-20c-no5	0.906	0.85	0.921	0.885	0.893	0.86	0.836	0.87	0.686	0.774	0.896
2d-20c-no6	0.942	0.869	0.943	0.947	0.95	0.868	0.862	0.921	0.677	0.819	0.921
2d-20c-no7	0.934	0.868	0.919	0.929	0.941	0.888	0.85	0.914	0.651	0.798	0.925
2d-20c-no8	0.943	0.855	0.933	0.943	0.946	0.878	0.911	0.91	0.58	0.786	0.906
2d-20c-no9	0.944	0.874	0.93	0.954	0.953	0.892	0.867	0.914	0.68	0.775	0.938
2d-40c-no0	0.939	0.875	0.944	0.943	0.944	0.856	0.87	0.905	0.798	0.8	0.919
2d-40c-no1	0.917	0.839	0.911	0.91	0.923	0.838	0.792	0.87	0.668	0.791	0.886
2d-40c-no2	0.898	0.83	0.896	0.902	0.904	0.82	0.804	0.859	0.653	0.793	0.871
2d-40c-no3	0.887	0.83	0.888	0.888	0.894	0.826	0.783	0.861	0.677	0.784	0.861
2d-40c-no4	0.945	0.873	0.93	0.928	0.929	0.874	0.815	0.921	0.724	0.798	0.908
2d-40c-no5	0.927	0.843	0.922	0.918	0.928	0.838	0.819	0.887	0.729	0.814	0.906
2d-40c-no6	0.92	0.861	0.933	0.922	0.931	0.815	0.769	0.89	0.753	0.811	0.91
2d-40c-no7	0.883	0.842	0.89	0.877	0.886	0.813	0.765	0.859	0.742	0.786	0.886
2d-40c-no8	0.899	0.85	0.93	0.895	0.907	0.843	0.755	0.875	0.652	0.828	0.887
2d-40c-no9	0.893	0.85	0.936	0.911	0.917	0.837	0.785	0.883	0.748	0.812	0.902
	0.926	0.888	0.944	0.934	0.935	0.881	0.783	0.921	0.712	0.721	0.933

Table 12 – Second type of experiments — Multi-modal — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD	MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	0.999	0.987	0.954	1	1	0.996	0.992	0.999	1	1	1	2d-4c-no0	0.975	0.85	0.733	0.974	0.982	0.951	0.235	0.972	0.65	0.62	0.927
2d-4c-no1	0.992	0.981	0.954	0.999	0.999	0.988	0.964	0.999	0.998	0.999	0.999	2d-4c-no1	0.701	0.79	0.69	0.966	0.969	0.903	0	0.969	0.567	0.64	0.877
2d-4c-no2	0.984	0.957	0.868	0.997	0.997	0.983	0.979	0.998	0.998	0.997	0.997	2d-4c-no2	0.526	0.709	0.651	0.738	0.785	0.867	0.22	0.927	0.616	0.571	0.706
2d-4c-no3	1	0.988	0.997	1	1	0.992	0.987	0.997	0.998	1	1	2d-4c-no3	0.971	0.833	0.953	0.968	0.97	0.869	0.587	0.942	0.541	0.492	0.983
2d-4c-no4	0.983	0.987	0.869	1	1	0.997	0.965	1	1	1	1	2d-4c-no4	0.62	0.837	0.344	0.956	0.966	0.938	0.394	0.964	0.484	0.337	0.928
2d-4c-no5	0.993	0.983	0.949	0.998	0.998	0.992	0.974	0.999	0.995	0.998	0.998	2d-4c-no5	0.803	0.747	0.513	0.936	0.938	0.856	0.176	0.949	0.507	0.731	0.775
2d-4c-no6	0.996	0.991	0.937	0.999	0.999	0.996	0.984	0.999	0.998	0.999	0.999	2d-4c-no6	0.941	0.852	0.634	0.961	0.961	0.944	0.247	0.973	0.655	0.526	0.89
2d-4c-no7	0.956	0.983	0.836	0.998	0.995	0.999	0.99	1	0.989	0.995	0.997	2d-4c-no7	0.524	0.736	0.454	0.956	0.883	0.959	0.226	0.967	0.619	0.415	0.92
2d-4c-no8	0.996	0.99	0.971	0.999	0.999	0.984	0.989	0.996	1	0.999	0.998	2d-4c-no8	0.936	0.751	0.711	0.955	0.948	0.79	0.089	0.934	0.574	0.601	0.858
2d-4c-no9	0.984	0.978	0.96	0.993	0.993	0.975	0.986	0.981	0.987	0.992	0.992	2d-4c-no9	0.707	0.759	0.685	0.728	0.723	0.7	0.242	0.73	0.49	0.299	0.874
2d-10c-no0	0.989	0.918	0.935	0.978	0.994	0.979	0.971	0.983	0.995	0.994	0.995	2d-10c-no0	0.776	0.632	0.368	0.813	0.835	0.779	0.015	0.836	0.506	0.592	0.707
2d-10c-no1	0.984	0.975	0.833	0.994	0.998	0.994	0.962	0.998	0.999	0.998	0.998	2d-10c-no1	0.596	0.564	0.166	0.846	0.912	0.876	0.009	0.916	0.573	0.47	0.747
2d-10c-no2	0.995	0.982	0.844	0.999	1	0.993	0.971	0.998	0.999	1	1	2d-10c-no2	0.858	0.641	0.201	0.934	0.935	0.859	0.008	0.892	0.525	0.448	0.807
2d-10c-no3	0.982	0.963	0.827	0.995	0.998	0.989	0.974	0.998	0.998	0.998	0.998	2d-10c-no3	0.367	0.63	0.392	0.863	0.901	0.86	0	0.903	0.58	0.528	0.682
2d-10c-no4	0.99	0.932	0.822	0.998	0.998	0.968	0.938	0.991	0.997	0.998	0.999	2d-10c-no4	0.702	0.539	0.273	0.926	0.937	0.59	0.051	0.711	0.584	0.633	0.652
2d-10c-no5	0.973	0.931	0.849	0.99	0.993	0.986	0.964	0.993	0.991	0.993	0.994	2d-10c-no5	0.249	0.48	0.06	0.658	0.759	0.657	-0.001	0.703	0.519	0.59	0.634
2d-10c-no6	0.952	0.852	0.814	0.992	0.994	0.981	0.961	0.994	0.99	0.994	0.994	2d-10c-no6	0.291	0.456	0.062	0.872	0.888	0.771	0.019	0.847	0.504	0.494	0.5
2d-10c-no7	0.982	0.93	0.817	0.986	0.995	0.984	0.974	0.995	0.995	0.993	0.995	2d-10c-no7	0.458	0.541	0.163	0.694	0.822	0.777	0.025	0.815	0.5	0.547	0.632
2d-10c-no8	0.917	0.907	0.821	0.988	0.988	0.966	0.952	0.992	0.986	0.987	0.99	2d-10c-no8	0.143	0.351	0.141	0.622	0.672	0.647	0.023	0.804	0.508	0.42	0.551
2d-10c-no9	0.986	0.942	0.835	0.995	0.999	0.98	0.954	0.994	0.997	0.998	0.999	2d-10c-no9	0.598	0.582	0.347	0.848	0.921	0.788	0.038	0.813	0.535	0.677	0.612
2d-20c-no0	0.998	0.934	0.773	0.998	0.999	0.989	0.979	0.998	1	1	1	2d-20c-no0	0.833	0.46	0.075	0.875	0.919	0.782	0.014	0.878	0.375	0.405	0.618
2d-20c-no1	0.987	0.942	0.815	0.992	0.999	0.988	0.957	0.996	1	0.999	0.999	2d-20c-no1	0.514	0.493	0.066	0.804	0.866	0.731	0.06	0.8	0.278	0.203	0.571
2d-20c-no2	0.972	0.916	0.804	0.991	0.994	0.971	0.947	0.992	0.998	0.995	0.995	2d-20c-no2	0.445	0.364	0.179	0.721	0.762	0.684	0.074	0.747	0.343	0.351	0.463
2d-20c-no3	0.994	0.936	0.793	0.991	0.997	0.983	0.968	0.995	0.999	0.997	0.999	2d-20c-no3	0.796	0.456	0.189	0.828	0.872	0.767	0.097	0.867	0.304	0.328	0.567
2d-20c-no4	0.99	0.922	0.79	0.994	0.998	0.98	0.949	0.995	0.999	0.998	0.998	2d-20c-no4	0.641	0.468	0.093	0.803	0.861	0.73	0.024	0.814	0.352	0.416	0.542
2d-20c-no5	0.977	0.913	0.795	0.987	0.992	0.966	0.944	0.988	0.986	0.992	0.993	2d-20c-no5	0.632	0.414	0.152	0.64	0.793	0.67	0.036	0.757	0.351	0.394	0.5
2d-20c-no6	0.978	0.891	0.801	0.983	0.989	0.978	0.955	0.993	0.998	0.99	0.993	2d-20c-no6	0.622	0.4	0.17	0.641	0.751	0.653	0.016	0.754	0.335	0.342	0.48
2d-20c-no7	0.988	0.921	0.824	0.992	0.995	0.974	0.96	0.986	0.994	0.994	0.995	2d-20c-no7	0.733	0.405	0.158	0.767	0.869	0.583	0.043	0.636	0.312	0.386	0.519
2d-20c-no8	0.984	0.939	0.78	0.993	0.995	0.988	0.936	0.992	0.995	0.996	0.997	2d-20c-no8	0.512	0.422	0.073	0.738	0.787	0.703	0	0.777	0.327	0.363	0.502
2d-20c-no9	0.989	0.911	0.834	0.995	0.998	0.988	0.971	0.996	0.999	0.998	0.998	2d-20c-no9	0.684	0.424	0.123	0.785	0.841	0.766	0	0.829	0.253	0.296	0.473
2d-40c-no0	0.978	0.789	0.774	0.993	0.995	0.969	0.949	0.992	0.995	0.995	0.997	2d-40c-no0	0.376	0.213	0.03	0.687	0.729	0.591	0	0.703	0.324	0.403	0.405
2d-40c-no1	0.973	0.808	0.773	0.989	0.994	0.961	0.927	0.983	0.995	0.994	0.992	2d-40c-no1	0.281	0.202	0.05	0.619	0.706	0.475	0.011	0.555	0.272	0.286	0.347
2d-40c-no2	0.953	0.761	0.786	0.978	0.984	0.949	0.895	0.976	0.98	0.984	0.984	2d-40c-no2	0.383	0.191	0.048	0.599	0.694	0.425	0.018	0.548	0.248	0.33	0.312
2d-40c-no3	0.969	0.808	0.82	0.982	0.991	0.954	0.938	0.979	0.993	0.991	0.991	2d-40c-no3	0.285	0.197	0.102	0.565	0.67	0.462	0.012	0.526	0.299	0.329	0.318
2d-40c-no4	0.977	0.816	0.784	0.989	0.995	0.975	0.933	0.992	0.995	0.995	0.997	2d-40c-no4	0.345	0.193	0.066	0.63	0.778	0.584	0.009	0.692	0.305	0.288	0.317
2d-40c-no5	0.97	0.745	0.802	0.98	0.988	0.959	0.924	0.986	0.991	0.988	0.989	2d-40c-no5	0.43	0.166	0.084	0.594	0.657	0.52	0.033	0.629	0.31	0.31	0.327
2d-40c-no6	0.971	0.758	0.756	0.99	0.993	0.959	0.935	0.988	0.995	0.993	0.993	2d-40c-no6	0.428	0.177	0.051	0.654	0.74	0.511	0	0.621	0.275	0.38	0.333
2d-40c-no7	0.957	0.768	0.797	0.984	0.985	0.966	0.933	0.989	0.994	0.986	0.99	2d-40c-no7	0.361	0.177	0.056	0.622	0.665	0.56	0.011	0.627	0.291	0.253	0.317
2d-40c-no8	0.966	0.761	0.807	0.986	0.988	0.959	0.913	0.983	0.989	0.989	0.99	2d-40c-no8	0.295	0.141	0.074	0.669	0.754	0.524	-0.001	0.636	0.25	0.283	0.302
2d-40c-no9	0.958	0.739	0.801	0.987	0.992	0.963	0.932	0.984	0.991	0.991	0.992	2d-40c-no9	0.207	0.176	0.027	0.575	0.69	0.486	0.025	0.567	0.311	0.309	0.308
	0.979	0.903	0.840	0.992	0.995	0.979	0.957	0.992	0.995	0.995	0.996		0.564	0.485	0.260	0.776	0.828	0.715	0.077	0.788	0.429	0.432	0.595

Table 13 – Third type of experiments — Local outliers — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	0.983	0.972	0.635	0.983	0.988	0.968	0.977	0.998	0.999	0.984	0.996
2d-4c-no1	0.922	0.862	0.493	0.975	0.948	0.914	0.897	0.957	0.974	0.958	0.973
2d-4c-no2	0.734	0.86	0.531	0.913	0.86	0.902	0.963	0.977	0.944	0.864	0.939
2d-4c-no3	0.772	0.855	0.512	0.987	0.874	0.978	0.936	0.989	0.967	0.903	0.967
2d-4c-no4	0.825	0.778	0.512	0.99	0.956	0.93	0.963	0.995	0.99	0.959	0.975
2d-4c-no5	0.963	0.886	0.589	0.986	0.99	0.954	0.902	0.993	0.977	0.981	0.993
2d-4c-no6	0.918	0.894	0.516	0.992	0.97	0.924	0.968	0.966	0.991	0.968	0.982
2d-4c-no7	0.867	0.881	0.681	0.97	0.928	0.955	0.932	0.935	0.975	0.926	0.955
2d-4c-no8	0.89	0.948	0.575	0.976	0.938	0.9	0.949	0.971	0.972	0.937	0.972
2d-4c-no9	0.866	0.914	0.689	0.955	0.86	0.943	0.93	0.973	0.964	0.865	0.979
2d-10c-no0	0.894	0.812	0.534	0.95	0.941	0.928	0.944	0.983	0.99	0.939	0.961
2d-10c-no1	0.845	0.79	0.495	0.925	0.895	0.88	0.884	0.889	0.895	0.893	0.947
2d-10c-no2	0.865	0.839	0.548	0.98	0.907	0.928	0.934	0.956	0.963	0.912	0.97
2d-10c-no3	0.865	0.785	0.557	0.97	0.919	0.92	0.882	0.971	0.949	0.928	0.97
2d-10c-no4	0.798	0.755	0.562	0.968	0.918	0.949	0.93	0.979	0.973	0.918	0.963
2d-10c-no5	0.813	0.74	0.491	0.967	0.918	0.924	0.949	0.963	0.965	0.919	0.957
2d-10c-no6	0.705	0.646	0.551	0.976	0.84	0.934	0.931	0.972	0.927	0.838	0.95
2d-10c-no7	0.863	0.815	0.662	0.928	0.886	0.911	0.842	0.959	0.875	0.892	0.919
2d-10c-no8	0.768	0.761	0.637	0.967	0.872	0.905	0.947	0.977	0.865	0.872	0.875
2d-10c-no9	0.812	0.815	0.605	0.974	0.919	0.941	0.905	0.984	0.97	0.922	0.964
2d-20c-no0	0.879	0.812	0.553	0.966	0.91	0.896	0.895	0.956	0.955	0.909	0.934
2d-20c-no1	0.874	0.831	0.67	0.931	0.92	0.942	0.942	0.958	0.964	0.928	0.973
2d-20c-no2	0.914	0.83	0.586	0.911	0.942	0.924	0.964	0.941	0.961	0.943	0.935
2d-20c-no3	0.819	0.785	0.48	0.944	0.886	0.932	0.817	0.954	0.845	0.871	0.936
2d-20c-no4	0.754	0.679	0.598	0.949	0.827	0.832	0.887	0.866	0.905	0.845	0.881
2d-20c-no5	0.846	0.899	0.641	0.949	0.911	0.927	0.984	0.946	0.988	0.904	0.919
2d-20c-no6	0.795	0.689	0.575	0.854	0.866	0.927	0.766	0.948	0.739	0.869	0.873
2d-20c-no7	0.629	0.568	0.45	0.946	0.72	0.878	0.891	0.938	0.799	0.739	0.801
2d-20c-no8	0.857	0.758	0.477	0.967	0.919	0.907	0.869	0.96	0.955	0.917	0.939
2d-20c-no9	0.891	0.819	0.57	0.965	0.922	0.928	0.914	0.965	0.973	0.931	0.972
2d-40c-no0	0.829	0.647	0.508	0.947	0.905	0.905	0.861	0.943	0.927	0.904	0.939
2d-40c-no1	0.741	0.649	0.524	0.921	0.817	0.898	0.92	0.939	0.804	0.822	0.853
2d-40c-no2	0.74	0.621	0.468	0.923	0.832	0.897	0.887	0.94	0.882	0.835	0.876
2d-40c-no3	0.789	0.653	0.503	0.932	0.849	0.888	0.846	0.911	0.867	0.861	0.918
2d-40c-no4	0.75	0.577	0.473	0.95	0.861	0.864	0.829	0.932	0.886	0.856	0.885
2d-40c-no5	0.839	0.699	0.559	0.923	0.879	0.847	0.74	0.937	0.891	0.887	0.923
2d-40c-no6	0.784	0.661	0.524	0.937	0.846	0.903	0.844	0.957	0.835	0.852	0.917
2d-40c-no7	0.83	0.667	0.414	0.914	0.876	0.925	0.905	0.933	0.912	0.884	0.937
2d-40c-no8	0.865	0.588	0.494	0.862	0.944	0.879	0.605	0.924	0.971	0.944	0.936
2d-40c-no9	0.817	0.683	0.411	0.927	0.897	0.887	0.869	0.949	0.972	0.895	0.91
	0.830	0.768	0.546	0.949	0.896	0.914	0.895	0.955	0.929	0.899	0.937

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	0.438	0.377	0.226	0.61	0.57	0.559	0.374	0.801	0.379	0.282	0.615
2d-4c-no1	0.451	0.292	0.061	0.594	0.408	0.341	0.608	0.535	0.222	0.255	0.278
2d-4c-no2	0.087	0.196	0.091	0.17	0.259	0.403	0	0.46	0.239	0.1	0.202
2d-4c-no3	0.047	0.232	0.049	0.238	0.301	0.429	0	0.675	0.384	0.21	0.338
2d-4c-no4	0.058	0.223	0.126	0.669	0.414	0.354	0	0.703	0.306	0.187	0.402
2d-4c-no5	0.558	0.338	0.209	0.599	0.748	0.646	-0.009	0.738	0.118	0.335	0.288
2d-4c-no6	0.223	0.466	0.094	0.678	0.547	0.445	0	0.675	0.239	0.148	0.315
2d-4c-no7	0.3	0.434	-0.016	0.516	0.567	0.497	0.443	0.667	0.252	0.126	0.299
2d-4c-no8	0.183	0.276	0.033	0.252	0.325	0.382	0.443	0.342	0.248	0.141	0.38
2d-4c-no9	0.556	0.341	0.106	0.34	0.431	-0.011	-0.012	-0.007	0.229	0.448	0.447
2d-10c-no0	0.33	0.215	0.101	0.177	0.356	0.235	0.286	0.363	0.432	0.18	0.233
2d-10c-no1	0.306	0.211	0.062	0.265	0.296	0.371	0	0.463	0.31	0.207	0.22
2d-10c-no2	0.226	0.197	0.108	0.609	0.414	0.462	0	0.633	0.274	0.058	0.388
2d-10c-no3	0.084	0.171	0.076	0.577	0.379	0.493	0	0.572	0.262	0.172	0.369
2d-10c-no4	0.201	0.177	0.16	0.383	0.461	0.529	0.165	0.539	0.382	0.094	0.311
2d-10c-no5	0.145	0.145	0.113	0.147	0.232	0.219	0.233	0.403	0.213	0.168	0.331
2d-10c-no6	0.235	0.116	0.137	0.316	0.348	0.384	0	0.511	0.218	0.083	0.258
2d-10c-no7	0.185	0.169	0.113	0.208	0.387	0.302	-0.007	0.417	0.198	0.108	0.217
2d-10c-no8	0.162	0.16	0.006	0.219	0.268	0.208	0	0.227	0.179	0.086	0.269
2d-10c-no9	0.282	0.343	0.08	0.174	0.315	0.343	0	0.555	0.326	0.165	0.305
2d-20c-no0	0.282	0.17	0.058	0.355	0.247	0.348	0	0.304	0.281	0.09	0.262
2d-20c-no1	0.081	0.201	0.029	0.246	0.308	0.473	0	0.492	0.278	0.158	0.241
2d-20c-no2	0.087	0.169	0.154	0.274	0.226	0.185	0.345	0.153	0.179	0.079	0.107
2d-20c-no3	0.29	0.188	0.193	0.281	0.494	0.351	0	0.49	0.181	0.09	0.21
2d-20c-no4	0.206	0.144	-0.017	0.271	0.227	0.297	-0.009	0.36	0.213	0.04	0.17
2d-20c-no5	0.41	0.384	0.133	0.134	0.302	0.245	0	0.374	0.353	0.056	0.212
2d-20c-no6	0.082	0.138	-0.005	0.089	0.285	0.19	0	0.38	0.184	0.076	0.203
2d-20c-no7	0.058	0.042	0.052	0.114	0.207	0.069	0	0.414	0.145	0.071	0.132
2d-20c-no8	0.109	0.118	-0.02	0.356	0.339	0.241	0	0.464	0.356	0.053	0.223
2d-20c-no9	0.29	0.184	-0.016	0.151	0.483	0.148	0	0.472	0.257	0.056	0.235
2d-40c-no0	0.238	0.098	-0.017	0.192	0.38	0.266	0	0.339	0.198	0.047	0.199
2d-40c-no1	0.048	0.087	0.077	0.134	0.221	0.249	0	0.345	0.15	0.082	0.168
2d-40c-no2	0.182	0.038	0.099	0.144	0.275	0.022	0	0.083	0.143	0.052	0.102
2d-40c-no3	0.157	0.069	0.033	0.171	0.156	0.238	0	0.152	0.195	0.134	0.22
2d-40c-no4	0.186	0.08	0.12	0.187	0.268	0.042	0.351	0.089	0.223	0.069	0.204
2d-40c-no5	0.262	0.135	0.067	0.309	0.35	0.109	0	0.075	0.137	0.036	0.212
2d-40c-no6	0.202	0.072	-0.013	0.225	0.286	0.224	0	0.294	0.157	0.062	0.181
2d-40c-no7	0.133	0.08	-0.015	0.225	0.207	0.288	0	0.231	0.169	0.09	0.19
2d-40c-no8	-0.013	0.039	-0.009	0.026	0.075	0.287	0	0.413	0.168	0.101	0.18
2d-40c-no9	0.05	0.046	-0.012	0.165	0.304	0.14	0	0.285	0.205	0.048	0.201
	0.210	0.189	0.071	0.295	0.342	0.300	0.080	0.412	0.240	0.126	0.258

Table 14 – Fourth type of experiments — Small sample size — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-20c-no0	1	0.97	1	1	1	1	1	1	1	1	1
2d-20c-no1	1	0.998	1	1	1	0.999	1	0.997	1	1	1
2d-20c-no2	0.998	0.989	0.994	0.998	0.998	0.995	1	0.998	1	0.998	0.998
2d-20c-no3	1	0.999	1	1	1	1	1	1	1	1	1
2d-20c-no4	1	0.998	1	1	1	1	1	1	1	1	0.999
2d-20c-no5	0.998	0.987	0.999	0.998	0.998	0.978	0.995	0.996	0.998	0.998	0.997
2d-20c-no6	0.996	0.992	0.997	0.995	0.996	0.995	0.999	0.996	1	0.996	0.996
2d-20c-no7	1	0.987	0.996	0.998	0.999	0.996	1	0.996	1	1	1
2d-20c-no8	1	0.99	0.995	0.998	0.999	0.996	0.994	0.997	0.998	0.998	0.999
2d-20c-no9	1	0.992	0.994	0.999	0.999	0.996	1	0.992	1	0.999	1
2d-40c-no0	1	0.998	0.999	1	1	0.998	1	0.999	1	1	0.999
2d-40c-no1	0.998	0.994	0.998	0.998	0.998	0.998	0.998	0.998	0.999	0.999	0.998
2d-40c-no2	0.999	0.995	0.998	0.999	0.999	0.997	0.999	0.998	0.998	0.999	0.999
2d-40c-no3	0.999	0.995	0.998	0.998	0.999	0.992	0.996	0.994	0.985	0.998	0.996
2d-40c-no4	1	0.994	0.998	0.999	0.999	0.999	0.999	0.997	0.999	0.999	0.999
2d-40c-no5	0.999	0.994	0.998	0.998	0.999	0.995	0.999	0.996	0.999	0.998	0.998
2d-40c-no6	0.999	0.994	0.997	0.999	0.999	0.996	0.999	0.998	1	0.999	0.999
2d-40c-no7	0.999	0.992	0.997	0.998	0.999	0.997	0.999	0.997	1	0.999	0.999
2d-40c-no8	0.998	0.993	0.998	0.998	0.999	0.998	0.998	0.999	0.998	0.998	0.998
2d-40c-no9	1	0.998	0.999	1	1	0.999	1	0.998	1	1	1
	0.999	0.992	0.998	0.999	0.999	0.996	0.999	0.997	0.999	0.999	0.999

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-20c-no0	0.917	0.741	0.957	0.906	0.906	0.861	0.668	0.799	0	0.957	0.917
2d-20c-no1	0.983	0.771	0.947	0.962	0.962	0.778	0.9	0.788	0.144	0.926	0.92
2d-20c-no2	0.806	0.649	0.775	0.83	0.833	0.738	0.785	0.78	0.345	0.872	0.799
2d-20c-no3	0.96	0.69	0.854	0.951	0.951	0.827	0.964	0.885	0.331	0.886	0.887
2d-20c-no4	0.892	0.794	0.866	0.942	0.952	0.803	0.813	0.902	0.463	0.902	0.904
2d-20c-no5	0.749	0.597	0.792	0.643	0.664	0.643	0.705	0.611	0.537	0.787	0.767
2d-20c-no6	0.85	0.666	0.816	0.853	0.85	0.763	0.789	0.798	0.485	0.798	0.781
2d-20c-no7	0.786	0.628	0.66	0.781	0.828	0.744	0.664	0.761	0.357	0.809	0.846
2d-20c-no8	0.85	0.638	0.784	0.852	0.86	0.738	0.865	0.798	0.371	0.823	0.722
2d-20c-no9	0.84	0.709	0.811	0.874	0.873	0.772	0.754	0.799	0.581	0.839	0.859
2d-40c-no0	0.907	0.751	0.881	0.907	0.916	0.75	0.906	0.81	0.614	0.898	0.785
2d-40c-no1	0.85	0.661	0.75	0.797	0.835	0.682	0.736	0.723	0.312	0.788	0.706
2d-40c-no2	0.821	0.66	0.738	0.806	0.816	0.695	0.775	0.691	0.333	0.824	0.811
2d-40c-no3	0.773	0.663	0.76	0.781	0.799	0.705	0.712	0.717	0.388	0.832	0.737
2d-40c-no4	0.878	0.752	0.777	0.83	0.847	0.775	0.811	0.789	0.561	0.865	0.764
2d-40c-no5	0.831	0.587	0.757	0.783	0.811	0.72	0.759	0.729	0.569	0.787	0.78
2d-40c-no6	0.781	0.715	0.792	0.785	0.81	0.61	0.662	0.713	0.592	0.871	0.824
2d-40c-no7	0.772	0.673	0.698	0.724	0.745	0.627	0.635	0.673	0.545	0.782	0.738
2d-40c-no8	0.847	0.768	0.876	0.821	0.857	0.769	0.763	0.763	0.416	0.856	0.795
2d-40c-no9	0.854	0.724	0.842	0.859	0.845	0.623	0.699	0.706	0.315	0.888	0.795
	0.847	0.692	0.807	0.834	0.848	0.731	0.768	0.762	0.413	0.850	0.807

Table 15 – Fifth type of experiments — Large sample size — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-4c-no1	0.999	0.997	1	0.999	0.999	0.997	0.998	0.999	0.997	0.999	0.999
2d-4c-no2	0.996	0.992	0.997	0.997	0.997	0.994	0.994	0.998	0.998	0.997	0.997
2d-4c-no3	1	0.998	1	1	1	0.998	0.996	0.999	0.999	1	1
2d-4c-no4	1	0.999	1	1	1	1	0.996	1	1	1	1
2d-4c-no5	0.999	0.996	0.999	0.999	0.999	0.997	0.998	0.999	0.992	0.999	0.999
2d-4c-no6	0.999	0.999	1	0.999	1	0.998	0.994	0.999	0.997	0.999	1
2d-4c-no7	0.997	0.997	1	0.998	0.999	0.998	0.996	0.999	0.995	0.999	0.999
2d-4c-no8	0.999	0.98	1	1	1	0.996	1	1	1	1	0.998
2d-4c-no9	0.996	0.996	0.997	0.998	0.998	0.995	0.994	0.998	0.997	0.997	0.999
2d-10c-no0	0.999	0.996	0.999	0.999	0.999	0.997	0.999	0.999	0.999	0.999	0.999
2d-10c-no1	0.999	0.999	1	1	1	0.999	0.998	1	1	1	1
2d-10c-no2	1	0.999	1	1	1	0.999	0.999	1	0.999	1	1
2d-10c-no3	0.999	0.998	1	1	1	0.999	0.998	1	0.999	0.999	1
2d-10c-no4	0.998	0.997	1	0.999	0.999	0.997	0.996	0.999	0.999	0.999	0.999
2d-10c-no5	0.987	0.983	0.991	0.989	0.991	0.983	0.988	0.99	0.984	0.99	0.989
2d-10c-no6	0.999	0.996	0.999	0.999	0.999	0.996	0.995	0.999	0.996	0.999	0.999
2d-10c-no7	0.998	0.997	1	0.999	0.999	0.997	0.998	0.999	0.999	0.998	0.999
2d-10c-no8	0.994	0.985	0.996	0.995	0.995	0.989	0.991	0.994	0.991	0.996	0.997
2d-10c-no9	0.999	0.992	0.999	0.999	0.999	0.995	0.998	0.999	0.998	0.999	0.999
2d-20c-no0	1	0.998	1	1	1	0.999	1	1	1	1	1
2d-20c-no1	1	1	1	1	1	1	1	1	1	1	1
2d-20c-no2	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-20c-no3	1	0.999	1	1	1	0.999	0.999	1	1	1	1
2d-20c-no4	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-20c-no5	0.999	0.998	0.999	0.999	0.999	0.998	0.999	0.999	0.999	0.999	0.999
2d-20c-no6	1	0.999	1	1	1	0.999	0.999	0.999	1	1	1
2d-20c-no7	1	0.999	1	1	1	0.999	0.999	0.999	0.999	1	0.999
2d-20c-no8	1	0.999	1	1	1	0.999	1	1	0.997	1	1
2d-20c-no9	1	1	1	1	1	1	1	1	1	1	1
2d-40c-no0	1	0.999	1	1	1	0.999	0.999	0.999	0.999	1	1
2d-40c-no1	1	0.999	1	1	1	0.999	0.999	1	1	1	1
2d-40c-no2	0.999	0.998	0.999	0.999	0.999	0.998	0.999	0.999	0.999	0.999	0.999
2d-40c-no3	1	0.998	1	1	1	0.999	0.999	0.999	1	1	1
2d-40c-no4	1	0.999	1	1	1	0.999	0.999	1	1	1	1
2d-40c-no5	1	0.998	1	1	1	0.999	0.999	0.999	1	1	1
2d-40c-no6	1	0.999	1	1	1	0.999	1	1	1	1	1
2d-40c-no7	1	0.999	1	1	1	0.998	1	0.999	1	1	1
2d-40c-no8	1	0.998	1	1	1	0.999	0.999	0.999	1	1	1
2d-40c-no9	1	0.998	1	1	1	0.999	0.999	0.999	0.999	1	1
	0.999	0.997	0.999	0.999	0.999	0.998	0.998	0.999	0.998	0.999	0.999

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
2d-4c-no0	0.988	0.94	0.991	0.988	0.99	0.971	0.897	0.983	0.772	0.727	0.983
2d-4c-no1	0.915	0.953	0.991	0.956	0.891	0.884	0.712	0.932	0.683	0.633	0.964
2d-4c-no2	0.875	0.88	0.926	0.898	0.887	0.855	0.697	0.878	0.731	0.646	0.931
2d-4c-no3	0.988	0.954	0.983	0.985	0.986	0.932	0.827	0.988	0.699	0.545	0.98
2d-4c-no4	0.996	0.876	0.991	0.991	0.994	0.964	0.75	0.986	0.46	0.614	0.97
2d-4c-no5	0.966	0.9	0.976	0.965	0.966	0.929	0.704	0.971	0.685	0.832	0.937
2d-4c-no6	0.987	0.934	0.984	0.985	0.986	0.971	0.812	0.985	0.701	0.648	0.986
2d-4c-no7	0.847	0.94	0.971	0.907	0.872	0.868	0.738	0.952	0.649	0.539	0.962
2d-4c-no8	0.874	0.853	0.961	0.933	0.937	0.839	0.746	0.957	0.57	0.604	0.966
2d-4c-no9	0.932	0.899	0.934	0.932	0.932	0.906	0.73	0.93	0.701	0.597	0.964
2d-10c-no0	0.978	0.937	0.965	0.976	0.977	0.953	0.869	0.967	0.822	0.644	0.962
2d-10c-no1	0.945	0.935	0.986	0.977	0.984	0.923	0.593	0.978	0.709	0.768	0.96
2d-10c-no2	0.988	0.933	0.992	0.99	0.993	0.933	0.79	0.983	0.779	0.598	0.978
2d-10c-no3	0.927	0.902	0.956	0.944	0.942	0.889	0.672	0.939	0.818	0.643	0.926
2d-10c-no4	0.876	0.928	0.969	0.919	0.909	0.902	0.761	0.926	0.868	0.612	0.969
2d-10c-no5	0.879	0.876	0.904	0.894	0.898	0.856	0.668	0.896	0.803	0.595	0.894
2d-10c-no6	0.87	0.904	0.922	0.881	0.884	0.843	0.589	0.88	0.805	0.633	0.942
2d-10c-no7	0.891	0.913	0.959	0.938	0.941	0.894	0.553	0.953	0.86	0.709	0.945
2d-10c-no8	0.856	0.839	0.864	0.868	0.871	0.822	0.788	0.864	0.818	0.74	0.913
2d-10c-no9	0.931	0.93	0.931	0.928	0.93	0.912	0.819	0.922	0.842	0.531	0.975
2d-20c-no0	0.97	0.958	0.983	0.963	0.969	0.91	0.915	0.95	0.735	0.763	0.956
2d-20c-no1	0.97	0.934	0.97	0.966	0.974	0.926	0.916	0.954	0.756	0.792	0.963
2d-20c-no2	0.958	0.924	0.975	0.969	0.971	0.911	0.846	0.942	0.723	0.787	0.957
2d-20c-no3	0.966	0.929	0.973	0.963	0.964	0.904	0.855	0.949	0.714	0.803	0.953
2d-20c-no4	0.936	0.907	0.962	0.946	0.95	0.91	0.821	0.936	0.742	0.786	0.946
2d-20c-no5	0.945	0.914	0.953	0.946	0.95	0.914	0.869	0.935	0.723	0.77	0.928
2d-20c-no6	0.965	0.92	0.975	0.97	0.975	0.894	0.88	0.952	0.725	0.824	0.957
2d-20c-no7	0.96	0.911	0.965	0.955	0.96	0.913	0.883	0.941	0.703	0.796	0.939
2d-20c-no8	0.974	0.928	0.982	0.974	0.975	0.925	0.927	0.947	0.65	0.774	0.967
2d-20c-no9	0.989	0.945	0.98	0.988	0.988	0.944	0.915	0.963	0.722	0.747	0.971
2d-40c-no0	0.947	0.906	0.96	0.952	0.951	0.883	0.861	0.929	0.844	0.776	0.953
2d-40c-no1	0.942	0.907	0.972	0.953	0.956	0.897	0.813	0.926	0.802	0.793	0.955
2d-40c-no2	0.921	0.879	0.942	0.93	0.929	0.857	0.812	0.908	0.746	0.784	0.888
2d-40c-no3	0.925	0.886	0.93	0.924	0.926	0.866	0.807	0.909	0.774	0.768	0.903
2d-40c-no4	0.961	0.903	0.969	0.952	0.95	0.899	0.816	0.954	0.765	0.781	0.944
2d-40c-no5	0.944	0.889	0.951	0.942	0.949	0.859	0.83	0.914	0.757	0.819	0.929
2d-40c-no6	0.955	0.898	0.968	0.956	0.961	0.867	0.796	0.934	0.793	0.796	0.931
2d-40c-no7	0.92	0.898	0.954	0.928	0.932	0.875	0.808	0.922	0.808	0.787	0.936
2d-40c-no8	0.914	0.874	0.945	0.916	0.922	0.864	0.753	0.908	0.721	0.82	0.913
2d-40c-no9	0.898	0.869	0.949	0.919	0.927	0.868	0.797	0.909	0.81	0.801	0.917
	0.937	0.910	0.960	0.947	0.946	0.898	0.791	0.939	0.745	0.716	0.948

Table 16 – Sixth type of experiments — High-Dimensional — Synthetic datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
20d-4c	1	1	1	1	1	1	1	1	1	1	1
20d-6c	1	1	1	1	1	1	1	1	1	1	1
22d-5c	1	1	1	1	1	1	1	1	1	1	1
22d-6c	1	1	1	1	1	1	1	1	0.999	1	1
22d-9c	1	1	1	1	1	1	1	1	1	1	1
23d-4c	1	1	1	1	1	1	1	1	0.999	1	1
23d-9c	1	1	1	1	1	1	1	1	1	1	1
24d-2c	1	1	1	1	1	1	1	1	1	1	1
24d-3c	1	1	1	1	1	1	1	1	1	1	1
24d-4c	1	1	1	1	1	1	1	1	1	1	1
24d-7c	1	1	1	1	1	1	1	1	1	1	1
25d-5c	1	1	1	1	1	1	1	1	1	1	1
25d-9c	1	1	1	1	1	1	1	1	0.999	1	1
26d-4c	1	1	1	1	1	1	1	1	0.999	1	1
27d-5c	1	1	1	1	1	1	1	1	1	1	1
27d-6c	1	1	1	1	1	1	1	1	1	1	1
28d-4c	1	1	1	1	1	1	1	1	1	1	1
28d-7c	1	1	1	1	1	1	1	1	1	1	1
29d-3c	1	1	1	1	1	1	1	1	1	1	1
30d-4c	1	1	1	1	1	1	1	1	1	1	1
31d-4c	1	1	1	1	1	1	1	1	1	1	1
33d-3c	1	1	1	1	1	1	1	1	1	1	1
33d-5c	1	1	1	1	1	1	1	1	0.999	1	1
35d-6c	1	1	1	1	1	1	1	1	1	1	1
36d-8c	1	1	1	1	1	1	1	1	1	1	1
36d-9c	1	1	1	1	1	1	1	1	0.999	1	1
37d-3c	1	1	1	1	1	1	1	1	0.999	1	1
38d-9c	1	1	1	1	1	1	1	1	1	1	1
39d-2c	1	1	1	1	1	1	1	1	1	1	1
39d-5c	1	1	1	1	1	1	1	1	0.999	1	1
	1	1	1	1	1	1	1	1	1	1	1

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
20d-4c	0.996	0.994	0.993	0.994	0.994	0.995	0.995	0.995	0	0.996	0.994
20d-6c	0.996	0.991	0.996	0.994	0.994	0.995	0.996	0.995	0	0.995	0.989
22d-5c	0.994	0.995	0.995	0.993	0.993	0.994	0.993	0.994	0	0.993	0.99
22d-6c	0.993	0.989	0.994	0.992	0.992	0.996	0.993	0.996	0	0.992	0.987
22d-9c	0.994	0.987	0.995	0.991	0.991	0.996	0.995	0.996	0	0.995	0.99
23d-4c	0.997	0.996	0.995	0.997	0.997	0.993	0.997	0.993	0	0.997	0.994
23d-9c	0.992	0.993	0.991	0.992	0.992	0.995	0.994	0.995	0	0.992	0.99
24d-2c	0.987	0.992	0.987	0.982	0.982	0.984	0.995	0.984	0	0.987	0.982
24d-3c	0.997	0.993	0.99	0.996	0.996	0.996	0.997	0.996	0	0.997	0.994
24d-4c	0.99	0.989	0.985	0.994	0.994	0.993	0.988	0.993	0	0.987	0.985
24d-7c	0.994	0.993	0.995	0.992	0.993	0.99	0.993	0.99	0	0.993	0.988
25d-5c	0.992	0.99	0.994	0.994	0.994	0.994	0.992	0.994	0	0.994	0.988
25d-9c	0.994	0.99	0.994	0.991	0.991	0.994	0.994	0.994	0	0.993	0.99
26d-4c	0.991	0.99	0.99	0.991	0.991	0.992	0.992	0.992	0	0.992	0.981
27d-5c	0.989	0.993	0.995	0.986	0.986	0.994	0.988	0.994	0	0.989	0.985
27d-6c	0.995	0.992	0.994	0.992	0.992	0.994	0.991	0.994	0	0.993	0.988
28d-4c	0.995	0.993	0.981	0.989	0.989	0.994	0.992	0.994	0	0.994	0.986
28d-7c	0.996	0.997	0.995	0.995	0.995	0.995	0.995	0.995	0	0.995	0.992
29d-3c	0.997	0.997	0.994	0.997	0.997	0.988	0.996	0.988	0	0.997	0.991
30d-4c	0.99	0.985	0.994	0.991	0.991	0.993	0.993	0.993	0	0.99	0.985
31d-4c	0.995	0.991	0.985	0.991	0.991	0.993	0.993	0.993	0	0.994	0.991
33d-3c	0.998	0.99	0.997	0.995	0.995	0.995	0.998	0.995	0	0.997	0.995
33d-5c	0.992	0.996	0.995	0.994	0.994	0.997	0.993	0.997	0	0.993	0.987
35d-6c	0.994	0.992	0.996	0.994	0.994	0.995	0.995	0.995	0	0.994	0.986
36d-8c	0.992	0.994	0.991	0.994	0.994	0.995	0.994	0.995	0	0.993	0.986
36d-9c	0.994	0.99	0.995	0.992	0.992	0.993	0.995	0.993	0	0.994	0.989
37d-3c	0.997	0.994	0.997	0.992	0.992	0.991	0.996	0.991	0	0.996	0.992
38d-9c	0.995	0.995	0.995	0.995	0.995	0.995	0.994	0.995	0	0.994	0.989
39d-2c	0.979	0.985	0.997	0.979	0.979	0.997	0.976	0.997	0	0.976	0.976
39d-5c	0.997	0.994	0.999	0.994	0.994	0.993	0.996	0.993	0	0.997	0.989
	0.993	0.992	0.993	0.992	0.992	0.994	0.993	0.994	0.000	0.993	0.988

Table 17 – First type of experiments — Unimodal — Real datasets

ROC AUC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
Abalone	0.71	0.72	0.74	0.66	0.73	0.66	0.76	0.66	0.71	0.74	0.77
Arrhythmia	0.51	0.53	0.55	0.63	0.52	0.58	0.53	0.6	0.5	0.5	0.64
Balance-Scale	0.86	0.91	0.93	0.88	0.87	0.86	0.87	0.92	0.88	0.87	0.91
Ball-Bearing	0.93	1	1	0.98	0.98	0.97	0.96	0.99	0.5	0.53	0.98
Biomed	0.65	0.72	0.8	0.84	0.84	0.81	0.79	0.71	0.51	0.69	0.7
Breast	0.93	0.91	0.98	0.96	0.96	0.93	0.98	0.96	0.81	0.8	0.98
Cancer	0.53	0.54	0.59	0.52	0.54	0.52	0.53	0.53	0.5	0.51	0.53
Colon	0.64	0.58	0.67	0.67	0.66	0.63	0.59	0.68	0.5	0.5	0.68
Delft	0.68	0.83	0.95	0.95	0.93	0.96	0.89	0.96	0.93	0.93	0.96
Diabetes	0.61	0.62	0.64	0.65	0.62	0.63	0.66	0.51	0.59	0.65	
Ecoli	0.93	0.89	0.94	0.94	0.94	0.93	0.94	0.94	0.92	0.94	0.94
Glass	0.79	0.76	0.78	0.78	0.81	0.78	0.67	0.81	0.81	0.81	0.82
Heart	0.59	0.67	0.73	0.6	0.58	0.57	0.58	0.59	0.5	0.55	0.61
Hepatitis	0.56	0.73	0.74	0.56	0.56	0.53	0.55	0.54	0.5	0.56	0.57
Housing	0.68	0.76	0.78	0.65	0.69	0.65	0.66	0.65	0.58	0.7	0.7
Imports	0.66	0.69	0.65	0.7	0.71	0.68	0.73	0.78	0.81	0.83	0.74
Ionosphere	0.64	0.62	0.64	0.74	0.67	0.66	0.6	0.64	0.66	0.64	0.74
Iris	0.97	0.96	0.98	0.97	0.97	0.95	0.97	0.97	0.98	0.98	0.98
Liver	0.55	0.54	0.54	0.54	0.55	0.55	0.58	0.55	0.53	0.53	0.56
Satellite	0.95	0.9	0.94	0.95	0.96	0.92	0.94	0.93	0.5	0.92	0.96
Sonar	0.64	0.67	0.66	0.7	0.76	0.73	0.65	0.77	0.76	0.76	0.75
Spectf	0.56	0.57	0.55	0.66	0.52	0.61	0.61	0.63	0.5	0.64	0.66
Survival	0.56	0.57	0.58	0.61	0.62	0.58	0.62	0.61	0.56	0.55	0.65
Vehicle	0.76	0.83	0.9	0.75	0.79	0.77	0.76	0.77	0.5	0.79	0.82
Vowels	0.99	0.63	0.99	0.99	1	0.98	0.91	0.99	1	1	0.99
Waveform	0.9	0.86	0.91	0.89	0.89	0.85	0.89	0.88	0.87	0.88	0.91
Wine	0.88	0.85	0.96	0.86	0.86	0.85	0.86	0.86	0.57	0.83	0.87
	0.73	0.74	0.78	0.76	0.76	0.75	0.74	0.76	0.66	0.72	0.78

MCC	ABOD	Auto Enc.	Gaussian	GLOSH	KNN G.	KNN L.	LOCI	LOF	LP	PW	SVDD
Abalone	0.04	0.03	0.01	0.03	0.03	0.03	0.08	0.03	0.2	0.11	0.33
Arrhythmia	0.06	0.08	0.18	0.09	0.07	0.06	0.07	0.08	0	0	0
Balance-Scale	0.46	0.5	0.51	0.43	0.42	0.4	0.15	0.5	0.27	0.47	0.45
Ball-Bearing	0.23	0.78	0.96	0.29	0.29	0.34	0.21	0.39	0	0	0.24
Biomed	0.21	0.2	0.28	0.2	0.2	0.19	0.21	0.2	0	0	0.2
Breast	-0.04	0.11	0.72	0.11	0.43	0.2	0.7	0.11	0.33	0.36	0.61
Cancer	0.01	0	0.06	0.05	0.01	0.05	0	0.02	0	0	0
Colon	0.11	0	0	0.07	0.04	0.05	-0.02	0.06	0	0	0.02
Delft	0.21	0.07	0.48	0.54	0.45	0.5	0.07	0.53	0	0	0.14
Diabetes	0	0.01	-0.01	0	0	0	0	0	0	0	0.12
Ecoli	0.23	0.2	0.29	0.26	0.26	0.28	0.21	0.24	0.51	0.37	0.49
Glass	0.16	0.15	0.33	0.15	0.16	0.15	0.09	0.16	0.28	0.23	0.24
heart	0	0.12	0.05	0	0	-0.01	-0.01	0.01	0	0	0.14
Hepatitis	-0.01	0.05	0.1	-0.02	-0.02	0	-0.01	-0.01	0	0	0.04
Housing	0.07	0.06	0.18	0.05	0.05	0.06	0.05	0.06	0	0	0.14
imports	0.01	0.02	0.2	0	-0.02	0.04	-0.01	0.2	0.07	0.04	0.29
Ionosphere	0.24	0.25	0.35	0.42	0.35	0.15	0.09	0.24	0.36	0.25	0.32
Iris	0.78	0.42	0.84	0.48	0.81	0.49	0.73	0.52	0.74	0.41	0.44
Liver	-0.06	0	0.02	0.02	0.02	0.01	0.01	0.01	0.06	0.08	0.1
Satellite	0.27	0.1	0.21	0.27	0.27	0.23	0.31	0.26	0	0	0.6
Sonar	0.05	0.26	0.24	0.16	0.16	0.09	0.02	0.13	0.08	0.02	0.02
Spectf	-0.3	0.13	0.17	0.31	0.15	0.15	0.14	0.16	0.33	0.52	0.13
Survival	-0.06	0	0.01	-0.01	0	0.01	0.02	0	0.06	0.11	0.07
Vehicle	0.15	0.25	0.29	0.2	0.21	0.23	0.16	0.25	0	0	0.32
Vowels	0.61	0.56	0.78	0.72	0.76	0.56	0.18	0.63	0.29	0.08	0.13
Waveform	0.42	0	0.44	0.48	0.48	0.41	0.19	0.49	0	0	0.56
Wine	0.47	0.3	0.69	0.44	0.45	0.49	0.41	0.45	0	0	0.3
	0.16	0.17	0.31	0.21	0.22	0.19	0.15	0.21	0.13	0.11	0.24

TABLES OF RESULTS, CHAPTER 4

The tables below show the detailed values for the results discussed in chapter 4. The tables display ROC AUC and MCC values for each method in the experiments performed in chapter 4. The highest achieved values for each data set are shown in bold.

Table 18 – First type of experiments — Unimodal — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
2d-4c-no0	1	1	1	2d-4c-no0	0.976	0.935	0.931
2d-4c-no1	0.998	0.998	0.998	2d-4c-no1	0.967	0.931	0.949
2d-4c-no2	0.995	0.995	0.997	2d-4c-no2	0.863	0.879	0.928
2d-4c-no3	1	1	0.999	2d-4c-no3	0.973	0.939	0.92
2d-4c-no4	1	1	1	2d-4c-no4	0.966	0.899	0.917
2d-4c-no5	0.999	0.999	0.999	2d-4c-no5	0.964	0.942	0.934
2d-4c-no6	1	1	1	2d-4c-no6	0.968	0.946	0.953
2d-4c-no7	0.999	0.999	0.999	2d-4c-no7	0.969	0.944	0.933
2d-4c-no8	0.999	0.997	1	2d-4c-no8	0.935	0.76	0.931
2d-4c-no9	0.997	0.991	0.997	2d-4c-no9	0.869	0.829	0.906
2d-10c-no0	0.998	0.999	0.998	2d-10c-no0	0.94	0.893	0.902
2d-10c-no1	1	1	0.999	2d-10c-no1	0.958	0.928	0.938
2d-10c-no2	1	1	1	2d-10c-no2	0.957	0.926	0.941
2d-10c-no3	0.999	0.999	0.999	2d-10c-no3	0.914	0.889	0.912
2d-10c-no4	0.999	0.999	0.999	2d-10c-no4	0.939	0.925	0.925
2d-10c-no5	0.988	0.984	0.987	2d-10c-no5	0.883	0.848	0.868
2d-10c-no6	0.999	0.999	0.999	2d-10c-no6	0.896	0.896	0.93
2d-10c-no7	0.999	0.999	0.999	2d-10c-no7	0.91	0.888	0.926
2d-10c-no8	0.995	0.994	0.995	2d-10c-no8	0.865	0.853	0.855
2d-10c-no9	0.998	0.998	0.999	2d-10c-no9	0.944	0.905	0.944
2d-20c-no0	1	1	1	2d-20c-no0	0.945	0.917	0.901
2d-20c-no1	1	1	1	2d-20c-no1	0.922	0.897	0.892
2d-20c-no2	0.999	0.999	0.999	2d-20c-no2	0.888	0.868	0.859
2d-20c-no3	1	1	1	2d-20c-no3	0.91	0.901	0.876
2d-20c-no4	1	1	1	2d-20c-no4	0.922	0.904	0.897
2d-20c-no5	0.999	0.999	0.999	2d-20c-no5	0.873	0.849	0.854
2d-20c-no6	0.999	0.999	0.999	2d-20c-no6	0.893	0.886	0.872
2d-20c-no7	0.999	0.999	0.998	2d-20c-no7	0.882	0.874	0.838
2d-20c-no8	1	1	1	2d-20c-no8	0.915	0.892	0.88
2d-20c-no9	1	1	0.999	2d-20c-no9	0.913	0.888	0.859
2d-40c-no0	1	1	1	2d-40c-no0	0.9	0.884	0.875
2d-40c-no1	0.999	0.999	0.999	2d-40c-no1	0.856	0.852	0.838
2d-40c-no2	0.998	0.999	0.998	2d-40c-no2	0.845	0.83	0.813
2d-40c-no3	0.999	0.999	0.999	2d-40c-no3	0.821	0.793	0.827
2d-40c-no4	1	1	1	2d-40c-no4	0.903	0.88	0.868
2d-40c-no5	1	1	0.999	2d-40c-no5	0.859	0.845	0.831
2d-40c-no6	0.999	1	0.999	2d-40c-no6	0.867	0.849	0.868
2d-40c-no7	0.999	1	0.999	2d-40c-no7	0.861	0.835	0.845
2d-40c-no8	0.999	0.999	0.999	2d-40c-no8	0.853	0.824	0.835
2d-40c-no9	0.999	0.999	0.999	2d-40c-no9	0.867	0.837	0.858
	0.999	0.999	0.999		0.909	0.882	0.891

Table 19 – Second type of experiments — Multi-modal — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
2d-4c-no0	0.999	0.999	0.999	2d-4c-no0	0.978	0.922	0.947
2d-4c-no1	0.998	0.998	0.997	2d-4c-no1	0.962	0.915	0.942
2d-4c-no2	0.986	0.988	0.995	2d-4c-no2	0.923	0.88	0.885
2d-4c-no3	0.999	0.999	1	2d-4c-no3	0.94	0.859	0.888
2d-4c-no4	1	1	0.998	2d-4c-no4	0.948	0.85	0.902
2d-4c-no5	0.997	0.998	0.998	2d-4c-no5	0.946	0.905	0.937
2d-4c-no6	0.999	0.999	0.999	2d-4c-no6	0.977	0.913	0.941
2d-4c-no7	0.995	0.994	0.986	2d-4c-no7	0.937	0.871	0.891
2d-4c-no8	0.998	0.998	0.999	2d-4c-no8	0.945	0.885	0.903
2d-4c-no9	0.967	0.954	0.988	2d-4c-no9	0.76	0.757	0.894
2d-10c-no0	0.947	0.969	0.986	2d-10c-no0	0.854	0.794	0.774
2d-10c-no1	0.997	0.997	0.997	2d-10c-no1	0.917	0.793	0.892
2d-10c-no2	1	0.999	0.999	2d-10c-no2	0.936	0.836	0.912
2d-10c-no3	0.99	0.996	0.996	2d-10c-no3	0.896	0.8	0.837
2d-10c-no4	0.998	0.998	0.998	2d-10c-no4	0.917	0.842	0.855
2d-10c-no5	0.987	0.986	0.99	2d-10c-no5	0.898	0.819	0.864
2d-10c-no6	0.97	0.988	0.991	2d-10c-no6	0.855	0.81	0.801
2d-10c-no7	0.994	0.993	0.994	2d-10c-no7	0.849	0.836	0.911
2d-10c-no8	0.936	0.988	0.969	2d-10c-no8	0.71	0.827	0.757
2d-10c-no9	0.99	0.994	0.996	2d-10c-no9	0.868	0.816	0.807
2d-20c-no0	1	0.999	0.999	2d-20c-no0	0.866	0.785	0.791
2d-20c-no1	0.998	0.998	0.998	2d-20c-no1	0.878	0.675	0.764
2d-20c-no2	0.991	0.993	0.993	2d-20c-no2	0.765	0.613	0.752
2d-20c-no3	0.996	0.997	0.996	2d-20c-no3	0.897	0.727	0.794
2d-20c-no4	0.993	0.995	0.994	2d-20c-no4	0.844	0.679	0.773
2d-20c-no5	0.982	0.986	0.987	2d-20c-no5	0.8	0.636	0.746
2d-20c-no6	0.975	0.984	0.982	2d-20c-no6	0.769	0.643	0.723
2d-20c-no7	0.989	0.992	0.992	2d-20c-no7	0.839	0.675	0.758
2d-20c-no8	0.995	0.996	0.996	2d-20c-no8	0.826	0.684	0.752
2d-20c-no9	0.997	0.998	0.996	2d-20c-no9	0.811	0.688	0.76
2d-40c-no0	0.993	0.993	0.992	2d-40c-no0	0.762	0.605	0.661
2d-40c-no1	0.99	0.99	0.989	2d-40c-no1	0.744	0.579	0.655
2d-40c-no2	0.975	0.969	0.97	2d-40c-no2	0.688	0.422	0.593
2d-40c-no3	0.982	0.982	0.985	2d-40c-no3	0.646	0.545	0.574
2d-40c-no4	0.992	0.993	0.993	2d-40c-no4	0.778	0.584	0.618
2d-40c-no5	0.981	0.982	0.98	2d-40c-no5	0.733	0.555	0.634
2d-40c-no6	0.989	0.991	0.99	2d-40c-no6	0.768	0.548	0.643
2d-40c-no7	0.97	0.982	0.974	2d-40c-no7	0.689	0.547	0.606
2d-40c-no8	0.984	0.985	0.981	2d-40c-no8	0.744	0.519	0.577
2d-40c-no9	0.967	0.987	0.983	2d-40c-no9	0.684	0.527	0.606
	0.987	0.991	0.991		0.839	0.729	0.783

Table 20 – Third type of experiments — Local outliers — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
2d-4c-no0	0.996	0.99	0.983	2d-4c-no0	0.889	0.701	0.735
2d-4c-no1	0.959	0.961	0.959	2d-4c-no1	0.705	0.617	0.637
2d-4c-no2	0.904	0.947	0.82	2d-4c-no2	0.624	0.529	0.243
2d-4c-no3	0.992	0.928	0.887	2d-4c-no3	0.816	0.501	0.335
2d-4c-no4	0.972	0.964	0.921	2d-4c-no4	0.71	0.576	0.47
2d-4c-no5	0.976	0.985	0.975	2d-4c-no5	0.793	0.692	0.726
2d-4c-no6	0.97	0.968	0.951	2d-4c-no6	0.705	0.657	0.582
2d-4c-no7	0.934	0.947	0.918	2d-4c-no7	0.636	0.56	0.47
2d-4c-no8	0.946	0.963	0.914	2d-4c-no8	0.626	0.651	0.473
2d-4c-no9	0.982	0.96	0.945	2d-4c-no9	0.78	0.623	0.65
2d-10c-no0	0.964	0.982	0.944	2d-10c-no0	0.744	0.68	0.564
2d-10c-no1	0.868	0.904	0.892	2d-10c-no1	0.563	0.52	0.456
2d-10c-no2	0.947	0.941	0.925	2d-10c-no2	0.751	0.587	0.628
2d-10c-no3	0.914	0.932	0.896	2d-10c-no3	0.709	0.554	0.485
2d-10c-no4	0.894	0.893	0.871	2d-10c-no4	0.61	0.507	0.482
2d-10c-no5	0.862	0.924	0.872	2d-10c-no5	0.537	0.538	0.4
2d-10c-no6	0.828	0.857	0.815	2d-10c-no6	0.543	0.45	0.375
2d-10c-no7	0.887	0.957	0.882	2d-10c-no7	0.53	0.626	0.57
2d-10c-no8	0.798	0.964	0.787	2d-10c-no8	0.426	0.667	0.224
2d-10c-no9	0.869	0.894	0.837	2d-10c-no9	0.62	0.449	0.316
2d-20c-no0	0.895	0.946	0.902	2d-20c-no0	0.476	0.551	0.474
2d-20c-no1	0.912	0.927	0.906	2d-20c-no1	0.529	0.455	0.463
2d-20c-no2	0.925	0.928	0.933	2d-20c-no2	0.492	0.392	0.501
2d-20c-no3	0.911	0.904	0.898	2d-20c-no3	0.527	0.509	0.542
2d-20c-no4	0.857	0.908	0.84	2d-20c-no4	0.532	0.425	0.459
2d-20c-no5	0.931	0.947	0.916	2d-20c-no5	0.598	0.494	0.54
2d-20c-no6	0.876	0.903	0.832	2d-20c-no6	0.459	0.367	0.321
2d-20c-no7	0.784	0.798	0.758	2d-20c-no7	0.419	0.307	0.352
2d-20c-no8	0.942	0.945	0.907	2d-20c-no8	0.604	0.539	0.543
2d-20c-no9	0.915	0.923	0.913	2d-20c-no9	0.563	0.497	0.525
2d-40c-no0	0.878	0.897	0.877	2d-40c-no0	0.458	0.439	0.443
2d-40c-no1	0.851	0.852	0.836	2d-40c-no1	0.366	0.344	0.327
2d-40c-no2	0.845	0.854	0.835	2d-40c-no2	0.398	0.313	0.378
2d-40c-no3	0.846	0.86	0.833	2d-40c-no3	0.4	0.377	0.346
2d-40c-no4	0.871	0.899	0.841	2d-40c-no4	0.408	0.392	0.354
2d-40c-no5	0.879	0.898	0.881	2d-40c-no5	0.481	0.447	0.451
2d-40c-no6	0.858	0.864	0.846	2d-40c-no6	0.375	0.36	0.358
2d-40c-no7	0.833	0.886	0.88	2d-40c-no7	0.453	0.428	0.425
2d-40c-no8	0.899	0.913	0.884	2d-40c-no8	0.356	0.376	0.379
2d-40c-no9	0.852	0.899	0.856	2d-40c-no9	0.438	0.433	0.426
	0.901	0.920	0.884		0.566	0.503	0.461

Table 21 – Fourth type of experiments — Small sample size — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
2d-20c-no0	1	1	1	2d-20c-no0	0.926	0.972	0.914
2d-20c-no1	1	1	0.999	2d-20c-no1	0.862	0.902	0.854
2d-20c-no2	0.996	0.999	0.997	2d-20c-no2	0.799	0.815	0.78
2d-20c-no3	1	1	1	2d-20c-no3	0.822	0.878	0.809
2d-20c-no4	1	1	1	2d-20c-no4	0.856	0.86	0.807
2d-20c-no5	0.998	0.999	0.999	2d-20c-no5	0.767	0.706	0.724
2d-20c-no6	0.994	0.996	0.994	2d-20c-no6	0.745	0.752	0.753
2d-20c-no7	0.997	0.999	0.992	2d-20c-no7	0.681	0.782	0.599
2d-20c-no8	0.999	1	0.999	2d-20c-no8	0.797	0.804	0.769
2d-20c-no9	0.999	0.999	0.998	2d-20c-no9	0.823	0.811	0.746
2d-40c-no0	1	1	0.999	2d-40c-no0	0.828	0.849	0.823
2d-40c-no1	0.997	0.998	0.996	2d-40c-no1	0.675	0.714	0.668
2d-40c-no2	0.998	0.998	0.998	2d-40c-no2	0.709	0.722	0.654
2d-40c-no3	0.998	0.999	0.998	2d-40c-no3	0.681	0.648	0.696
2d-40c-no4	0.999	0.999	0.999	2d-40c-no4	0.804	0.756	0.772
2d-40c-no5	0.999	0.999	0.997	2d-40c-no5	0.615	0.642	0.576
2d-40c-no6	0.998	0.999	0.999	2d-40c-no6	0.68	0.7	0.752
2d-40c-no7	0.999	0.999	0.998	2d-40c-no7	0.685	0.623	0.694
2d-40c-no8	0.998	0.997	0.997	2d-40c-no8	0.744	0.72	0.728
2d-40c-no9	0.999	1	1	2d-40c-no9	0.788	0.732	0.782
	0.998	0.999	0.998		0.764	0.769	0.745

Table 22 – Fifth type of experiments — Large sample size — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
2d-4c-no0	1	1	1	2d-4c-no0	0.976	0.935	0.931
2d-4c-no1	0.998	0.998	0.998	2d-4c-no1	0.967	0.931	0.949
2d-4c-no2	0.995	0.995	0.997	2d-4c-no2	0.863	0.879	0.928
2d-4c-no3	1	1	0.999	2d-4c-no3	0.973	0.939	0.92
2d-4c-no4	1	1	1	2d-4c-no4	0.966	0.899	0.917
2d-4c-no5	0.999	0.999	0.999	2d-4c-no5	0.964	0.942	0.934
2d-4c-no6	1	1	1	2d-4c-no6	0.968	0.946	0.953
2d-4c-no7	0.999	0.999	0.999	2d-4c-no7	0.969	0.944	0.933
2d-4c-no8	0.999	0.997	1	2d-4c-no8	0.935	0.76	0.931
2d-4c-no9	0.997	0.991	0.997	2d-4c-no9	0.869	0.829	0.906
2d-10c-no0	0.998	0.999	0.998	2d-10c-no0	0.94	0.893	0.902
2d-10c-no1	1	1	0.999	2d-10c-no1	0.958	0.928	0.938
2d-10c-no2	1	1	1	2d-10c-no2	0.957	0.926	0.941
2d-10c-no3	0.999	0.999	0.999	2d-10c-no3	0.914	0.889	0.912
2d-10c-no4	0.999	0.999	0.999	2d-10c-no4	0.939	0.925	0.925
2d-10c-no5	0.988	0.984	0.987	2d-10c-no5	0.883	0.848	0.868
2d-10c-no6	0.999	0.999	0.999	2d-10c-no6	0.896	0.896	0.93
2d-10c-no7	0.999	0.999	0.999	2d-10c-no7	0.91	0.888	0.926
2d-10c-no8	0.995	0.994	0.995	2d-10c-no8	0.865	0.853	0.855
2d-10c-no9	0.998	0.998	0.999	2d-10c-no9	0.944	0.905	0.944
2d-20c-no0	1	1	1	2d-20c-no0	0.946	0.914	0.901
2d-20c-no1	1	1	1	2d-20c-no1	0.937	0.896	0.901
2d-20c-no2	1	1	1	2d-20c-no2	0.926	0.891	0.892
2d-20c-no3	1	1	1	2d-20c-no3	0.926	0.905	0.888
2d-20c-no4	1	1	1	2d-20c-no4	0.934	0.911	0.913
2d-20c-no5	0.999	0.999	0.999	2d-20c-no5	0.9	0.884	0.886
2d-20c-no6	1	1	1	2d-20c-no6	0.929	0.919	0.901
2d-20c-no7	0.999	1	0.999	2d-20c-no7	0.918	0.89	0.88
2d-20c-no8	1	1	1	2d-20c-no8	0.954	0.923	0.92
2d-20c-no9	1	1	1	2d-20c-no9	0.952	0.921	0.908
2d-40c-no0	1	1	1	2d-40c-no0	0.918	0.893	0.888
2d-40c-no1	1	1	1	2d-40c-no1	0.925	0.904	0.902
2d-40c-no2	0.999	0.999	0.999	2d-40c-no2	0.884	0.86	0.861
2d-40c-no3	0.999	1	0.999	2d-40c-no3	0.867	0.841	0.871
2d-40c-no4	1	1	1	2d-40c-no4	0.928	0.911	0.892
2d-40c-no5	1	1	0.999	2d-40c-no5	0.905	0.885	0.88
2d-40c-no6	1	1	1	2d-40c-no6	0.914	0.886	0.897
2d-40c-no7	1	1	1	2d-40c-no7	0.92	0.906	0.895
2d-40c-no8	0.999	0.999	0.999	2d-40c-no8	0.885	0.854	0.866
2d-40c-no9	0.999	0.999	0.999	2d-40c-no9	0.878	0.858	0.869
	0.999	0.999	0.999		0.925	0.895	0.906

Table 23 – Sixth type of experiments — High-dimensional — Synthetic datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
20d-4c	1	1	1	20d-4c	0.977	0.933	0.994
20d-6c	1	1	1	20d-6c	0.974	0.927	0.993
22d-5c	1	1	1	22d-5c	0.977	0.921	0.993
22d-6c	1	1	1	22d-6c	0.982	0.93	0.994
22d-9c	1	1	1	22d-9c	0.98	0.925	0.993
23d-4c	1	1	1	23d-4c	0.974	0.93	0.994
23d-9c	1	1	1	23d-9c	0.975	0.935	0.994
24d-2c	1	1	1	24d-2c	0.974	0.906	0.993
24d-3c	1	1	1	24d-3c	0.975	0.921	0.994
24d-4c	1	1	1	24d-4c	0.974	0.926	0.994
24d-7c	1	1	1	24d-7c	0.974	0.927	0.994
25d-5c	1	1	1	25d-5c	0.977	0.922	0.993
25d-9c	1	1	1	25d-9c	0.981	0.922	0.994
26d-4c	1	1	1	26d-4c	0.991	0.917	0.992
27d-5c	1	1	1	27d-5c	0.983	0.921	0.993
27d-6c	1	1	1	27d-6c	0.985	0.922	0.993
28d-4c	1	1	1	28d-4c	0.991	0.924	0.993
28d-7c	1	1	1	28d-7c	0.989	0.93	0.994
29d-3c	1	1	1	29d-3c	0.993	0.932	0.993
30d-4c	1	1	1	30d-4c	0.988	0.92	0.992
31d-4c	1	1	1	31d-4c	0.99	0.922	0.994
33d-3c	1	1	1	33d-3c	0.981	0.915	0.995
33d-5c	1	1	1	33d-5c	0.969	0.925	0.993
35d-6c	1	1	1	35d-6c	0.976	0.918	0.993
36d-8c	1	1	1	36d-8c	0.979	0.924	0.994
36d-9c	1	1	1	36d-9c	0.983	0.921	0.995
37d-3c	1	1	1	37d-3c	0.971	0.919	0.992
38d-9c	1	1	1	38d-9c	0.978	0.926	0.994
39d-2c	1	1	1	39d-2c	0.983	0.894	0.992
39d-5c	1	1	1	39d-5c	0.981	0.921	0.994
	1	1	1		0.980	0.923	0.993

Table 24 – First type of experiments — Unimodal — Real datasets

ROC AUC	Perturbation	SDS	Uniform Objects	MCC	Perturbation	SDS	Uniform Objects
Abalone	0.661	0.673	0.661	Abalone	0.071	0.166	0.071
Arrhythmia	0.513	0.505	0.505	Arrhythmia	0.068	0.027	0.019
Balance-Scale	0.877	0.822	0.855	Balance-Scale	0.479	0.483	0.477
Ball-Bearing	0.989	0.978	0.964	Ball-Bearing	0.518	0.718	0.2
Biomed	0.716	0.731	0.721	Biomed	0.255	0.251	0.265
Breast	0.978	0.943	0.886	Breast	0.772	0.727	0.38
Cancer	0.484	0.509	0.484	Cancer	-0.051	-0.052	0.013
Colon	0.639	0.732	0.651	Colon	0.126	0.088	0.088
Delft	0.918	0.946	0.833	Delft	0.377	0.371	0.159
Diabetes	0.642	0.65	0.649	Diabetes	0.011	0.175	0.055
Ecoli	0.902	0.91	0.902	Ecoli	0.231	0.319	0.231
Glass	0.719	0.712	0.755	Glass	0.301	0.181	0.186
Heart	0.581	0.544	0.53	Heart	0.033	0.033	0.015
Hepatitis	0.636	0.596	0.579	Hepatitis	0.118	0.08	0
Housing	0.66	0.741	0.692	Housing	0.054	0.173	0.03
Ionosphere	0.648	0.645	0.677	Ionosphere	0.187	0.295	0.125
Iris	0.984	0.983	0.982	Iris	0.765	0.68	0.758
Liver	0.546	0.594	0.569	Liver	0.058	0.045	0.039
Satellite	0.936	0.947	0.935	Satellite	0.251	0.542	0.237
Sonar	0.633	0.548	0.621	Sonar	0.113	0.093	0.135
Spambase	0.707	0.707	0.612	Spambase	0.061	0.102	0.074
Spectf	0.485	0.522	0.607	Spectf	0.078	-0.004	0.047
Survival	0.605	0.558	0.657	Survival	0.106	0.1	0.121
Vehicle	0.726	0.778	0.753	Vehicle	0.155	0.289	0.093
Vowels	0.993	0.997	0.986	Vowels	0.767	0.748	0.608
Waveform	0.881	0.868	0.879	Waveform	0.338	0.379	0.333
Wine	0.839	0.824	0.859	Wine	0.299	0.295	0.271
	0.737	0.739	0.733		0.242	0.271	0.186