UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Semi-supervised learning approaches with applications in medicinal chemistry

Jadson Castro Gertrudes

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: _____

Jadson Castro Gertrudes

Semi-supervised learning approaches with applications in medicinal chemistry

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Ricardo José Gabrielli Barreto Campello

USP – São Carlos
July 2019

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

C355s Castro Gertrudes, Jadson
 Semi-supervised learning approaches with
 applications in medicinal chemistry / Jadson Castro
 Gertrudes; orientador Ricardo José Gabrielli
 Barreto Campello. -- São Carlos, 2019.
 133 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. Semi-supervised clustering. 2. Semi-
 supervised classification. 3. density-based
 clustering. 4. structure-activity relationship. I.
 José Gabrielli Barreto Campello, Ricardo , orient.
 II. Título.

Jadson Castro Gertrudes

**Abordagens de aprendizado semissupervisionado com
aplicações em química medicinal**

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
Julho de 2019**

ACKNOWLEDGEMENTS

A minha querida família, meus pais Josenilza e Gonçalo, meus irmãos Jardel e Josiney e minha esposa Valéria, que sempre me apoiaram durante toda esta caminhada. Sem todo o suporte dado por eles não teria chegado tão longe. Muito obrigado por existirem em minha vida.

Ao meu orientador Ricardo J. G. B. Campello por seu papel fundamental no meu crescimento profissional e pessoal. Agradeço pela sua persistência em sempre buscar excelência nos nossos trabalhos. Também agradeço por todas as dicas dadas durante o doutorado. Vou levar todos estes ensinamentos para o resto dos meus dias.

Ao professor Jörg Sander pela recepção e pelo acompanhamento durante os 11 meses do doutorado sanduíche realizado na Universidade de Alberta, no Canadá. Também agradeço o professor Arthur Zimek por todas as dicas dadas durante o doutorado. As contribuições deles foram fundamentais para o desenvolvimento da tese.

A Roberto, Daniel, Rafael Mantovani, Diego Damasceno, Elias, Geraldo, George, Eri-naldo, Maria, Lúcio, Silvana, Daniela Ridel, Rodolfo, Faimison, Victor Barella, Victor Padilha, Victor Hugo, Arthur, Henrique, Brauner, Stevão, Francisco, Raphael Montanari, Jefferson, Everlândio, Prof. Adenilso, Leo e a todos os amigos de São Carlos. Meus agradecimentos por todos os momentos que passei por lá.

A todos os amigos que fiz durante o estágio no Canadá. Antônio (Toni), Camila, Marjorie, Mariana, Gustavo Coutinho, Victor, Leandro, Rodolfo, Priscila, Sydnie, Garry, Omid e tantos outros. Agradeço pelas discussões científicas e pelos momentos de diversão.

Sou grato também aos membros do comitê avaliador por seu tempo, suas considerações e contribuições ao trabalho. Elas serão refletidas na versão final do texto.

A todos os funcionários do ICMC-USP (professores, seguranças, membros da secretaria, etc.) por todo o suporte dado durante o meu doutorado.

Por fim, agradeço a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo auxílio financeiro durante o doutorado e ao CNPq pelo suporte financeiro durante o estágio de pesquisa no exterior.

ABSTRACT

GERTRUDES, J. C. **Semi-supervised learning approaches with applications in medicinal chemistry**. 2019. 133 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.

Semi-supervised learning is drawing increasing attention in the era of big data, as the gap between the abundance of cheap, automatically collected unlabeled data and the scarcity of labeled data that are laborious and expensive to obtain is dramatically increasing. In this thesis, we first introduce a unified view of density-based clustering algorithms. Then, we build upon this view and bridge the areas of semi-supervised clustering and classification under a common umbrella of density-based techniques. We show that there are close relations between density-based clustering algorithms and the graph-based approach for transductive classification. These relations are then used as a basis for a new framework for semi-supervised classification based on building-blocks from density-based clustering. This framework is not only efficient and effective, but it is also statistically sound. We also generalize the core algorithm of the framework HDBSCAN* so that it can also perform semi-supervised clustering by directly taking advantage of any fraction of labeled data that may be available, rather than instance-level pairwise constraints. Experimental results on a large collection of datasets show the advantages of the proposed approach both for semi-supervised classification, as well as for semi-supervised clustering. In addition, we evaluate the semi-supervised learning algorithms to determine relationships between chemical structure and biological activity in datasets from Medicinal Chemistry. The datasets evaluated in this area are characterized by a low number of labeled examples, a high dimensionality, and in some cases, do not have a clear relationship between chemical structure and biological activity, which makes it difficult to use classification techniques and analyze chemical phenomena. We implement and validate semi-supervised classification approaches that are appropriate for data analysis in Medicinal Chemistry.

Keywords: Semi-supervised clustering, semi-supervised classification, density-based clustering, structure-activity relationship.

RESUMO

GERTRUDES, J. C. **Abordagens de aprendizado semissupervisionado com aplicações em química medicinal**. 2019. 133 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.

O aprendizado semissupervisionado obteve, no decorrer do tempo, cada vez mais significância na área de *big data*, uma vez que a lacuna entre a abundância de dados não rotulados coletados de forma rápida e barata e a escassez de dados rotulados que são trabalhosos e caros de obter, está aumentando drasticamente. No presente documento, apresentamos uma visão unificada dos algoritmos de agrupamento baseados em densidade. Depois, utilizamos essa visão unificada para construção de uma “ponte” entre as áreas de agrupamento e classificação semissupervisionadas. Mostramos que existem relações entre algoritmos de agrupamento baseados em densidade e a abordagem baseada em grafos para classificação transdutiva. Essas relações são então usadas como base para um novo *framework* de classificação semissupervisionada com base em algoritmos baseados em densidade. Assim, definimos um novo *framework* que pode ser dividido em pequenos blocos de construção, com base nos algoritmos baseado em densidade. Essa estrutura é eficiente, eficaz e também estatisticamente sólida. Também generalizamos o *framework* HDBSCAN* para que ele também pudesse realizar agrupamento semissupervisionado utilizando diretamente qualquer fração de dados rotulados que possam estar disponíveis, ao invés de um conjunto de restrições. Resultados experimentais em uma grande coleção de base de dados mostram as vantagens da abordagem proposta tanto para classificação semissupervisionada quanto para o agrupamento semissupervisionado. Além disso, avaliamos os algoritmos de aprendizado semissupervisionados para determinar as relações entre a estrutura química e a atividade biológica em conjuntos de dados da área de Química Medicinal. Os conjuntos de dados avaliados nesta área são caracterizados por um baixo número de exemplos rotulados, alta dimensionalidade e, em alguns casos, não possuem uma relação clara entre estrutura química e atividade biológica, o que dificulta a aplicação de técnicas aprendizado supervisionado. Após implementação e validação das abordagens semissupervisionadas, verificamos que os mesmos podem ser uma boa opção para análise de dados em Química Medicinal.

Palavras-chave: Agrupamento semissupervisionado, Classificação Semissupervisionada, agrupamento baseado em densidade, Análise de Relação entre Estrutura Química e Atividade Biológica.

LIST OF FIGURES

Figure 1 – Tree showing traditional clustering algorithms.	30
Figure 2 – Illustrative data set.	34
Figure 3 – Dendrogram obtained from the illustrative data set (Figure 2), using the Euclidean Distance and the Single Linkage clustering.	35
Figure 4 – Core point.	36
Figure 5 – Reachability-plot for a two dimensional data set. Valleys in the reachability plot may indicate the presence of clusters on the data.	38
Figure 6 – Mutual reachability distance for objects $\mathbf{x}_p$ and $\mathbf{x}_q$	39
Figure 7 – Cluster tree generated from the hierarchy shown in Table 2.	42
Figure 8 – Cluster tree and unsupervised FOSSC extraction with stability.	44
Figure 9 – A graph composed from both labeled ($\mathbf{x}_1$ and $\mathbf{x}_2$ and unlabeled objects. The unlabeled object $\mathbf{x}_3$ will obtain the label from $\mathbf{x}_1$ even though it is closer (i.e., in Euclidean distance) from $\mathbf{x}_2$	51
Figure 10 – SSDBSCAN label assignment: in case (I), object $\mathbf{x}_i$ is assigned to $\mathbf{x}_{j2}$ (red); in case (II), $\mathbf{x}_i$ is left unlabeled as noise.	55
Figure 11 – Label assignment in HISSCLU: object $\mathbf{x}_2$ is assigned to $\mathbf{x}_1$ (green), whereas $\mathbf{x}_3$ and $\mathbf{x}_4$ are assigned to $\mathbf{x}_5$ (red).	58
Figure 12 – Drug discovery and development process.	61
Figure 13 – Possible interactions between a bioactive substance and its biological target.	62
Figure 14 – Example of density-based label expansion.	69
Figure 15 – Unified framework of density-based algorithms for semi-supervised classification.	71
Figure 16 – Quality of results obtained by the semi-supervised classification algorithms averaged over all datasets, separated by different percentage of labeled objects.	77
Figure 17 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test.	77
Figure 18 – Runtime over dataset size.	78
Figure 19 – B^3 Precision and B^3 Recall of an object $\mathbf{x}_j$. Objects within the ellipse belong to the same cluster. Red crosses and blue stars represent two different class labels. Unlabeled objects are represented by black filled circles. Notice that any two objects with the same class label in the same cluster have the same precision and recall.	82

Figure 20 – HDBSCAN* cluster tree for the hierarchy in Table 4 and flat clustering extraction using FOSC: (a) Unsupervised extraction using Stability; (b) — (d) Labeled-based semi-supervised extraction using B^3 Precision, Recall, and F-Measure, respectively; (e) Mixed case. The set of pre-labeled objects, $\mathbf{X}_L$, is represented using colored symbols: $\mathbf{x}_1$ and $\mathbf{x}_6$ as red crosses, $\mathbf{x}_8$ as a green hash, $\mathbf{x}_{15}$ as a magenta triangle, and $\mathbf{x}_{18}$ as a blue star. Singleton nodes corresponding to noise objects that are not pre-labeled are omitted as they do not affect computations. Extracted clusters in each case are highlighted in bold.	84
Figure 21 – ARI results for the Artificial collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	93
Figure 22 – ARI results for the Artificial collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	94
Figure 23 – ARI results for the ALOI collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	95
Figure 24 – ARI results for the ALOI collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	96
Figure 25 – ARI results for the Real collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	97
Figure 26 – ARI results for the Real collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.	98
Figure 27 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: Artificial collection (non-controlled random scenario).	99
Figure 28 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: ALOI collection (non-controlled random scenario).	99
Figure 29 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: Real collection (non-controlled random scenario).	99
Figure 30 – Runtime over dataset size.	100
Figure 31 – Diagram for the first framework specified for SAR analysis. The steps involve using of a label propagation algorithm before the application of the variable selection method.	108

Figure 32 – Summary for the experiments presented in the section, which refers to the label propagation and feature selection using the propagated labels.	108
Figure 33 – Diagram for the second framework defined for SAR analysis. The steps involve using of a feature selection method before the application of the label propagation algorithm.	112
Figure 34 – Summary for the experiments presented in the section, which refers to the feature selection and label propagation procedures.	112
Figure 35 – Diagram for the third framework defined for SAR analysis. The framework combines both semi-supervised clustering and feature selection techniques for label propagation and selection of relevant features.	116
Figure 36 – Summary for the experiments presented in the section, which refers to the feature selection and label propagation procedures.	116

LIST OF TABLES

Table 1	– Cluster hierarchy with $m_{\text{pts}} = m_{\text{CISize}} = 3$ obtained from the illustrative dataset presented in Figure 2. The first line contains the labels of all the objects in the dataset, and the column ϵ presents the density level where one object becomes part of a cluster or a noise.	41
Table 2	– Compact hierarchy with $m_{\text{pts}} = m_{\text{CISize}} = 3$ obtained from the illustrative dataset presented in Figure 2.	41
Table 3	– List of real datasets collected to perform the semi-supervised classification experiments.	73
Table 4	– Cluster hierarchy, extracted from the illustrative dataset presented in Figure 2, with $m_{\text{pts}} = m_{\text{CISize}} = 3$. Colors (red, blue, green, and magenta) highlight a collection of pre-labeled objects, $\mathbf{X}_L = \{\mathbf{x}_1, \mathbf{x}_6, \mathbf{x}_8, \mathbf{x}_{15}, \mathbf{x}_{18}\}$, which can be used to extract clusters from the hierarchy in a semi-supervised way (cf. Figure 20).	85
Table 5	– List of datasets collected to perform the semi-supervised clustering experiments.	88
Table 6	– ARI results (mean $\pm$ standard deviation) of the original SSDBSCAN implementation and our improved implementation, SSDBSCAN++, in the non-controlled random scenario with different amounts of pre-labeled objects.	92
Table 7	– List of chemical datasets collected to perform the semi-supervised classification experiments.	104
Table 8	– Average results (accuracy) obtained for each dataset presented in Table 7, using the label propagation and feature selection. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors green , to represent an improvement, and red to represent a drop.	109
Table 9	– Average results (accuracy) obtained for each dataset presented in Table 7, considering the framework presented in Figure 34. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors green , to represent an improvement, and red to represent a drop in performance.	113

Table 10 – Average results (accuracy) obtained for each dataset presented in Table 7, using the feature selection combined with the SSDBSCAN++ algorithm. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors **green**, to represent an improvement, and **red** to represent a drop in performance. 117

LIST OF ABBREVIATIONS AND ACRONYMS

AL	Average-Linkage
CL	Complete-Linkage
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
FOSC	Framework for Optimal Extraction of Clusters
FS	Fisher Score
GFHF	Gaussian Field Harmonic Function
HDBSCAN*	Hierarchical DBSCAN*
HISSCLU	Hierarchical Density-Based Method for Semi-Supervised Clustering
LapSVM	Laplacian Support Vector Machines
LSDF	Locality Sensitive Discriminant Feature
MC	Medicinal Chemistry
OPTICS	Ordering Points To Identify the Clustering Structure
RMGT	Robust Multi-Class Graph Transduction
SAR	Structure-activity Relationship
SL	Single-Linkage
SSDBSCAN	Semi-supervised DBSCAN
SSE	Sum of Squared Errors
WM	Ward's Method

CONTENTS

1	INTRODUCTION	23
1.1	Contributions	25
1.2	Notations	26
1.3	Outline	26
2	CLUSTER ANALYSIS	29
2.1	Partitional algorithms	31
2.2	Hierarchical algorithms	32
2.3	Density-based clustering	35
2.3.1	<i>DBSCAN*, DBSCAN, and OPTICS</i>	35
2.3.2	<i>HDBSCAN*</i>	38
2.3.2.1	<i>HDBSCAN* with all-points core distance</i>	42
2.4	FOSC	43
2.5	Final remarks	47
3	SEMI-SUPERVISED LEARNING	49
3.1	SSDBSCAN	53
3.1.1	<i>Conceptual Approach</i>	53
3.1.2	<i>Algorithmic Approach</i>	55
3.1.3	<i>Shortcomings</i>	56
3.2	HISSCLU	57
3.2.1	<i>Preprocessing stage (label-based distance weighting)</i>	57
3.2.2	<i>Label expansion stage</i>	58
3.2.3	<i>Flat Clustering Extraction (k-clustering)</i>	59
3.3	Final remarks	59
4	COMPUTER-AIDED DRUG DESIGN	61
4.1	SAR in Medicinal Chemistry	63
4.2	Final remarks	64
5	UNIFIED FRAMEWORK FOR DENSITY-BASED CLASSIFICATION	65
5.1	The Components of the Framework (Building Blocks)	65
5.2	The Framework	70
5.3	Complexity	72

5.4	Experimental setup	72
5.4.1	<i>Performance measure</i>	73
5.4.2	<i>Datasets</i>	74
5.4.3	<i>Pre-labeled data subsets</i>	74
5.4.4	<i>Algorithms and parameters</i>	74
5.4.5	<i>Statistical test for performance comparison</i>	75
5.4.6	<i>Runtime experiments</i>	76
5.5	Results	76
5.5.1	<i>Effectiveness</i>	76
5.5.2	<i>Efficiency</i>	78
5.6	Final remarks	79
6	LABEL BASED SEMI-SUPERVISED FOSC	81
6.1	Label-based quality criteria for optimal cluster extraction	81
6.2	Complexity	86
6.3	Experimental setup	87
6.3.1	<i>Performance measures</i>	87
6.3.2	<i>Datasets</i>	87
6.3.3	<i>Pre-labeled data subsets</i>	89
6.3.4	<i>Algorithms and Parameters</i>	90
6.3.5	<i>Statistical test and runtime experiments</i>	91
6.4	Results	91
6.4.1	<i>Improved SSDBSCAN</i>	91
6.4.2	<i>Effectiveness</i>	91
6.4.3	<i>Efficiency</i>	100
6.5	Final remarks	100
7	APPLYING SEMI-SUPERVISED LEARNING FOR STRUCTURE- ACTIVITY ANALYSIS	103
7.1	Experimental setup	103
7.1.1	<i>Datasets</i>	104
7.1.2	<i>Data partitioning and partial information</i>	105
7.1.3	<i>Feature selection</i>	105
7.1.4	<i>Algorithms and parameters</i>	107
7.1.5	<i>Validation</i>	107
7.1.6	<i>Source code</i>	107
7.2	Propagating labels for feature selection	108
7.2.1	<i>Results</i>	109
7.3	Selecting features for label propagation	111
7.3.1	<i>Results</i>	112

7.4	Combining Feature selection and SSDBSCAN++	115
7.4.1	<i>Results</i>	116
7.5	Final remarks	119
8	CONCLUSIONS AND FUTURE WORK	121
8.1	Future work	122
8.2	Publications and financial support	124
BIBLIOGRAPHY		127

INTRODUCTION

Semi-supervised learning algorithms address cases where a relatively small amount of labeled data yet a large amount of unlabeled data is available for training (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; ZHU; GOLDBERG, 2009). We find examples of semi-supervised learning scenarios in various fields, such as email filtering, sound/speech recognition, text/webpage classification, and compound discovery, just to mention a few. For instance, in areas such as biology, chemistry, and medicine, domain experts and laboratory analyses may be required to label observations, thus only a small collection of labeled data can usually be afforded, which may not be representative enough for supervised learning to be applied (BATISTA; CAMPELLO; SANDER, 2016).

In general, semi-supervised learning algorithms are based on extensions of either supervised or unsupervised algorithms by including additional information in the form originally handled by the other learning paradigm. For instance, in semi-supervised *classification*, the classes are known in advance, and unlabeled training data are used in addition to the labeled ones to improve on classification performance. In semi-supervised *clustering*, a collection of labeled observations can be used to guide the search for clustering solutions that better meet users' earlier expectations. Labels in semi-supervised clustering show whether observations are expected to be part of the same cluster or different clusters. There is not a direct association between the unique labels known to the user and the possible categories to be discovered in the data.

Semi-supervised learning can be categorized into *inductive* and *transductive* learning. Inductive learning uses both labeled and unlabeled observations to generate a model able to predict the labels for the unlabeled ones, as well as for future data to be labeled. Transductive learning predicts the labels of the unlabeled training data only, that may be used to derive a model able to predict new data (not used during the transduction). In transductive *classification*, which will be the first focus of this document, a large amount of unlabeled objects can be classified based on a small fraction of labeled objects, which is not large enough to successfully train a

classifier in a fully supervised way. Label-based semi-supervised *clustering*, the second focus of this thesis, is intrinsically transductive in its nature. It diverges from transductive classification because in semi-supervised clustering not all possible categories and their labels are necessarily known in advance, so unlabeled objects may be assigned to newly discovered labels that are not present in the original training set.

Relying on semi-supervised classification, it can be observed, according to [Zhu \(2005\)](#), that unlabeled data can help improve classification performance depending on a good matching of problem structure with *model assumption*. Different models of semi-supervised classification exist, based on different model assumption. One of the major paradigms, clustering-based models, follows the *cluster assumption* of the data, which is quite general and broad in scope as there are many possible interpretations of a “cluster”, under different clustering paradigms. Besides the cluster assumption, many algorithms also consider the *smoothness assumption* of the data important for semi-supervised classification, which considers that objects with the same label should share the same high-density region while being separated from other objects by a low-density-area.

An important paradigm that has been surprisingly overlooked in the literature is *density-based clustering* ([KRIEGEL et al., 2011](#)) that has strong connections with the semi-supervised classification area, specially regarding the cluster and smooth-based based methods. Many semi-supervised classification methods focus instead on using graphs (as opposed to clusters) to model the notions of locality and connectivity of the data ([SOUSA; REZENDE; BATISTA, 2013](#)). These graph-based methods mostly rely on the *graph assumption* of the data.

The lack of studies on density-based clustering methods in semi-supervised classification literature is the first line of investigation in this thesis. We show that there is a strong relationship between density-based clustering methods and the graph-based approach for transductive classification, by first establishing formal relationships between a number of key unsupervised and semi-supervised clustering algorithms under a unified view of density-based clustering, then establishing the links and interpretations of these algorithms from the perspective of the graph theory. Based on this unified view, we then introduce a framework of density-based clustering for semi-supervised classification that brings the assumptions mentioned above under a common umbrella.

We also extend our unified view of density-based methods from the semi-supervised classification scenario to the label-based semi-supervised *clustering* scenario, where labels for certain categories may be missing in the training set. As a novel contribution to this context, we extend HDBSCAN*, which plays a central role in our unified view and framework for density-based classification, to also perform semi-supervised clustering from a collection of pre-labeled data objects, rather than instance-level pairwise constraints (as currently supported by the algorithm). The direct use of labels can be shown to be both simpler and more effective.

Besides the introduction of the unified view of density-based methods for semi-supervised classification and the label-based semi-supervised clustering, we also present a study that consists of using semi-supervised learning methods for compound mining. The study is related to the Medicinal Chemistry area, which aims to investigate the main mechanisms related to the development of a disease to propose new candidate molecules for medicines. Researchers from this area are responsible for validating a biological target, discovering or designing new compounds, conducting biological tests, among other activities (YOUNG, 2009).

Currently, machine learning and data mining techniques are being used widespread in Medicinal Chemistry, especially classification/regression techniques, as they help reduce financial and time costs during the drug discovery process (YOUNG, 2009; MALTAROLLO, 2013). We present in this document the analysis of relationships between chemical structure and biological activities, called structure-activity relationships (SAR). In SAR, a database of molecules is used that already has information about their interaction with the biological target (active and inactive), whereby multivariate data analysis methods are applied to establish the relationships between the chemical structure and its respective biological activity.

One of the main difficulties of applying classification/regression techniques for SAR is the low number of labeled molecules. In general, most chemicals do not have pre-defined labels and samples of labeled compounds are not large enough to generate a good classification/regression model. In general, only the labeled data are used to build the learning model. The unlabeled information could increase the performance of learning algorithms. For this reason, we developed a set of algorithms for SAR analysis and evaluate them in a controlled scenario.

1.1 Contributions

The contribution of this thesis is summarized as follows:

- In the context of semi-supervised classification
 - Our framework extends the state-of-the-art density-based hierarchical clustering algorithm HDBSCAN* (CAMPELLO *et al.*, 2015) originally proposed as an unsupervised or (constraint-based) semi-supervised clustering algorithm, to perform transductive *classification* from a small collection of pre-labeled data objects;
 - We show that, in the context of transductive classification, other well-known density-based algorithms for semi-supervised clustering can also be derived as particular cases, with the advantage that our framework eliminates possible order-dependency and graph re-computation issues of these algorithms, while being simpler and easier to interpret;

- By combining building blocks from different algorithms, a number of novel variants follow naturally from our framework, which, to the best of our knowledge, has never been tried before.
- In the context of semi-supervised clustering, we extend HDBSCAN*, which plays a central role in our unified view and framework for density-based classification, to also perform semi-supervised clustering from a collection of pre-labeled data objects, rather than instance-level pairwise constraints (as currently supported by the algorithm). The developed approach can be shown to be both simpler and more effective.
- Lastly, we also present as a contribution a set of algorithms for compound mining based on semi-supervised learning algorithms, that shows to be more effective than using only the the set of labeled objects provided by domain experts for SAR analysis.

1.2 Notations

In this document, we adopt the following notations: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ is a dataset with n data objects, $\mathbf{x}_i$. Some of the algorithms described in this paper assume that data objects are points in a d -dimensional Euclidean space, i.e., $\mathbf{x}_i \in \mathbb{R}^d$ is a d -dimensional feature vector with real-valued coordinates ($\mathbf{x}_i = [x_{i1} \dots x_{id}]^T$). Others do not make any assumptions about features and only require a measure of dissimilarity between pairs of data objects, $d(\mathbf{x}_i, \mathbf{x}_j)$, in order to operate. This dissimilarity is assumed to be a distance but not necessarily a metric. $\mathbf{X}_L \subset \mathbf{X}$ is a subset of the data objects for which class labels are available, and $\text{class}(\mathbf{x}_i)$ is the class label of object $\mathbf{x}_i \in \mathbf{X}_L$. The subset of unlabeled objects is denoted by $\mathbf{X}_U$, such that $\mathbf{X}_L \cup \mathbf{X}_U = \mathbf{X}$ and $\mathbf{X}_U = \mathbf{X} \setminus \mathbf{X}_L$.

1.3 Outline

The thesis is organized in 8 chapters. The remainder of the document is organized as follows:

- [Chapter 2](#) we present the basic concepts of cluster analysis, such as partitional and hierarchical algorithms. We also present a detailed description of density-based clustering algorithms, foundation of the main contributions of this document.
- [Chapter 3](#) presents concepts of semi supervised learning, such as semi-supervised classification and semi-supervised clustering. We also present two main algorithms defined as semi-supervised clustering algorithms that are, in fact, semi-supervised classification algorithms. We also identify some shortcomings of these algorithms, that are enhanced in one of our contributions in this thesis.

- [Chapter 4](#) presents a description of compound mining activity, performed in the Medicinal Chemistry area. One of the activities consists of analyzing relationships between the chemical structure and biological activity. We also describe some semi-supervised learning algorithms already applied in this area.
- [Chapter 5](#) shows the first contribution of this document. We present a novel framework for semi-supervised classification, where the bedrock is density-based clustering algorithms. The novel framework showed to be faster and presented promising results in a diverse set of public available data.
- [Chapter 6](#) describes another important contribution of the thesis. We present a novel label-based measure for optimal cluster extraction from hierarchies. The proposed method avoids using constraints, commonly used in a semi-supervised clustering scenario.
- [Chapter 7](#) presents some studies that we performed using the analysis between the chemical structure and biological activity using different semi-supervised frameworks.
- In [Chapter 8](#), we present the conclusions of this thesis, as well as the main contributions published during the doctorate.

CLUSTER ANALYSIS

Cluster analysis (also known as data clustering) is one of the principal conceptual issues both in the Machine Learning ([MITCHELL, 1997](#); [BISHOP, 2006](#)) and in Data Mining areas ([HAN, 2005](#); [TAN](#); [STEINBACH](#); [KUMAR, 2006](#); [THEODORIDIS](#); [KOUTROUMBAS, 2009](#)). It is characterized by the search for a categorization that shows the relationships between the objects (instances, observations, examples, etc.) of a set of data of interest. Such data sets are commonly represented by attribute vectors (feature vectors) that describe all objects from their characteristics, by quantifying their similarities from measures of (dis) similarity or by graphs that model the inter-relationship between objects ([JAIN](#); [DUBES, 1988](#); [KAUFMAN](#); [ROUSSEEUV, 2005](#); [AGGARWAL](#); [REDDY, 2013](#)).

In many practical applications, the solution to this problem is often the ultimate goal of the analysis procedure, however, in many other cases solving a clustering problem it is only the first step in the problem solving process such as pattern classification, the extraction of rules or the categorization of texts. For its wide applicability in most varied areas of knowledge, such as biology, economics, medicine, chemistry, sociology, etc., and for gathering elements from different disciplines such as statistics, algorithms, pattern recognition and artificial intelligence, clustering is considered as an interdisciplinary field of research ([HARTIGAN, 1975](#); [JAIN](#); [DUBES, 1988](#)).

Although there is no exact and globally accepted cluster definition, over the last few decades several clustering methods have been created and used to find and analyze cluster patterns in data from several areas ([JAIN](#); [DUBES, 1988](#); [XU](#); [Wunsch-II, 2009](#)). Since the categories to which the objects can be assigned are not known a priori, the data grouping procedure is considered as an unsupervised type of machine learning. Its methods can be divided into two main types, the partitional methods and the hierarchical methods, as presented in [Figure 1](#). The partitioning algorithms produce a single partitioning of the data as output. The solutions produced by the hierarchical methods have the shape of a tree that describes the relationship between nested groups. This type of solution is especially interesting in a number of application

areas because (i) there is indeed a hierarchy in the data set investigated; (ii) in some areas, such as in biology, specialists may prefer a hierarchical representation because they are more familiar with this type of representation; (iii) a representation with different levels is inherent to the domain of the application, as in the categorization of documents; (iv) there are useful tools for visualizing and analyzing clustering solutions at different levels of granularity (JAIN; DUBES, 1988; EVERIT; LANDAU; LEESSE, 2001).

Figure 1 – Tree showing traditional clustering algorithms.

Source: Elaborated by the author.

Basically, the clustering problem can be defined as the search for k groups to which the N objects of the data set should be assigned so that the objects belonging to each group are more similar or interrelated to each other than the other groups. Partial algorithms can still be subdivided according to the type of partition they produce: unique partitioning algorithms (hard, not overlapping) assign each object to only one group, while non-unique partitioning algorithms (soft, with overlapping) define the degree in which each object belongs to a group. Following Figure 1, the hierarchical algorithms can, according to the way they operate, be subdivided into agglomerative and divisive ones. Agglomerative algorithms begin their operation by assigning to each object a different group and, at each step, choose two (or more) more similar groups to unite until all objects belong to a single group. In the case of the divisive algorithms, the process is inverse. From a single group containing all objects, at each step a group is chosen to be divided into two (or more) subgroups until no other group can be divided. In both cases, heuristics can be defined that interrupt the process before its completion if they consider the rest of the groups yet to be created as unnecessary (KAUFMAN; ROUSSEUW, 2005; XU; Wunsch-II, 2009).

In cases where a flat partition is also desired (where each object is assigned to only a cluster) it can be extracted from the hierarchy by choosing (either manually or automatically) one of its levels. The choice of how to cut a hierarchy horizontally is a classic problem that has been investigated for decades (JAIN; DUBES, 1988). Although widely used, horizontal cuts

have considerable limitations that prevent the choice of solutions composed of groups identified at different levels of abstraction. This problem particularly affects hierarchical density-based algorithms in which hierarchical nesting of groups and subgroups is characterized by density differences and is directly associated with the definition of [Hartigan \(1975\)](#) of density-contour trees. In this paradigm, a horizontal cut represents the choice of a single threshold of density global ([KRIEGEL et al., 2011](#)), which also represents one of the major shortcomings of many algorithms based on non-hierarchical density, such as DBSCAN ([ESTER et al., 1996](#)) and DENCLUE ([HINNEBURG; KEIM et al., 1998](#)).

2.1 Partitional algorithms

Partitional clustering algorithms seek to categorize objects belonging to a set of data of interest into disjoint subsets. For example, it can be assumed that in a data set composed by the register of movies visualized by N different users, there are k user groups that have similar movie interests in such a way that all N users can be assigned to one of these k groups. In a formal way, consider $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ the data set containing N objects $(1, 2, \dots, n)$. Each object $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}]$ is composed of d distinct variables.

A partition $\mathbf{P} = \{\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_k\}$ of this data set, consisting of k groups, is a collection of subsets of objects $\mathbf{C}_j$ ($j = 1, \dots, k$), such that:

- $\mathbf{C}_1 \cup \mathbf{C}_2 \cup \dots \cup \mathbf{C}_k = \mathbf{X}$;
- $\forall j, \mathbf{C}_j \neq \emptyset$;
- $\forall i \neq j, \mathbf{C}_i \cap \mathbf{C}_j = \emptyset$.

In other words: (i) the union of all subsets (groups, *clusters*) is equal to data set $\mathbf{X}$, that is, it is necessary to assign one object to a group; (ii) each group $\mathbf{C}_j$ is non-empty, i.e. at least one object is assigned to each group; (iii) the intersection of two groups $\mathbf{C}_i$ and $\mathbf{C}_j$ is empty, which means that one object can be assigned to only one cluster.

Assuming k is known, the number of possible ways to group N objects in k clusters is given by:

$$\frac{1}{k!} \sum_{i=0}^k (-1)^i \binom{k}{i} (k-i)^N \quad (2.1)$$

Using $N = 100$ and $k = 5$ as an example, there are approximately 56.6×10^{67} possible ways to partition $\mathbf{X}$. Given that k is generally unknown, the problem is even greater, and it is impossible to enumerate and evaluate all partitions. For this reason, over the last decades,

different algorithms have been proposed that seek to produce good partitions. Among them, the k-means algorithm is probably the best known of all data-mining algorithms (JAIN, 2010).

The idea behind the k-Means algorithm has been discussed since the 1950s (although the most cited publication dates back to 1967 (MACQUEEN, 1967)). In its original form, the algorithm assumes that all n variables describing the data set are real, i.e. each object $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}]$ is a numeric vector that can be understood as a point in a d -dimensional space.

Assuming there are k groups in $\mathbf{X}$, the k-means algorithm creates k prototypes and tries to fit them so that each one of them represents, in the best possible way, one of the k groups present in the data. This means that the prototype of a cluster should be the most similar point to the objects belonging to it. This similarity is defined as the sum of the square (Euclidean) distances between this point and the objects belonging to a cluster. Thus, the goal of k-means is to select k points (prototypes) $\bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_k$, where the Sum of Squared Errors (SSE) from these points to the objects of their respective clusters is minimized. To solve this problem, k-means uses an iterative approach consisting of two main steps: (i) assigning the objects to one of the prototypes; and (ii) updating these prototypes. In general terms, the algorithm is defined by the following steps: (a) choose k prototypes randomly; (b) each object is assigned to the nearest prototype; (c) the prototypes of each cluster are recalculated as the multivariate mean of the objects of that cluster; (d) repeat steps (b) and (c) until no other object changes from cluster (or until a given stop criterion is reached).

At each step, it is guaranteed that the algorithm does not increase the SSE so that it eventually converges to a solution with a minimum SSE. Although it always converges to a minimum, there is no guarantee that the value reached will be the global minimum. Thus, the results achieved can vary significantly, depending on the choice of the initial prototypes. In addition, k-means has two other critical problems such as the choice of the number of groups k (which in practice is unknown) and the use of prototypes that induce globular groups (generally incapable of detecting groups with arbitrary shapes). Hierarchical algorithms, presented in the following section, do not have a problem of the number of groups, and some of them also do not have a problem with the globular shapes. Although they experience from other problems that will be described later.

2.2 Hierarchical algorithms

Unlike partitional algorithms, where a non-hierarchical partition (flat partition) is produced, the hierarchical clustering algorithms, which is also presented in Figure 1, produce a collection composed of several solutions with varying numbers of groups. In this collection, the number of groups in each partition ranges from $k = 1$ (a single group containing all objects) to $k = N$ (N groups, each with only one object). In addition, there is a relationship between the

partitions that compose it so that the partition $\mathbf{P}_k$, composed of k different groups, is contained in the partition $\mathbf{P}_{k-1}$, consisting of $k-1$ groups. A partition $\mathbf{P}_k$ is contained in another partition $\mathbf{P}_l$ if, any group $\mathbf{C}_i$ belonging to $\mathbf{P}_k$ is a subset of some group $\mathbf{C}_j$ belonging to $\mathbf{P}_l$. For example, let $\mathbf{P}_5 = (\{\mathbf{x}_1\}, \{\mathbf{x}_2, \mathbf{x}_4\}, \{\mathbf{x}_5, \mathbf{x}_6\}, \{\mathbf{x}_3, \mathbf{x}_7\})$ and $\mathbf{P}_4 = (\{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_7\}, \{\mathbf{x}_2, \mathbf{x}_4\}, \{\mathbf{x}_5, \mathbf{x}_6\})$, then $\mathbf{P}_5 \subset \mathbf{P}_4$. In this case, the partitions are said to be nested, and such a collection is called a hierarchy.

Hierarchical clustering algorithms can build these hierarchies in agglomerative fashion (starting from $\mathbf{P}_N$ and joining the most similar clusters two-by-two) or in a divisive way (starting from $\mathbf{P}_1$ and choosing a cluster to be split). In practice, agglomerative algorithms are better known and used because they are simpler and more intuitive. The following are the main steps of a hierarchical agglomerative algorithm:

1. Initially each object is considered as an individual cluster (i.e., a singleton);
2. The most similar pair of clusters is attached, according to a given measure of (dis) similarity between clusters;
3. Repeat step 2 until there remains only a single cluster containing all objects in the data set.

Note that in the second step, as two clusters are joined, the total number of clusters is reduced by one unit, i.e., partition $\mathbf{P}_k$ which has k clusters is transformed into the partition $\mathbf{P}_{k-1}$ which contains $k-1$ clusters. Thus, the algorithm produces N partitions with $k = N, N-1, \dots, 1$ clusters, respectively.

In order for the description of a generic agglomerative algorithm, as explained above, to be complete, it is necessary to specify how the (dis) similarity between groups is calculated. Basically it is this choice that differentiates the different types of agglomerative algorithms. Among them, the most used in practice are Single-Linkage (SL), Complete-Linkage (CL), Average-Linkage (AL) and Ward's Method (WM).

The hierarchical algorithms described above have different positive and negative points: SL is the only algorithm capable of identifying arbitrary format clusters, the others favor globular clusters (such as the clusters provided by k-Means). However, SL is very sensitive to noise, especially the background noise that can create artificial bridges between the groups, thus causing undesired joining of these groups. The CL prevents this effect but is sensitive to the presence of outliers, which drastically distort the greater distance between groups (measure used by CL) causing the breakdown of very large clusters. AL and WM are generally more stable and reliable, although they cannot detect clusters of arbitrary formats.

Regardless of the hierarchical algorithm used, the hierarchy produced can be viewed by means of a dendrogram, as shown in [Figure 2](#). A dendrogram can be understood as an “inverted tree”, where objects are placed on the bottom (leaves), groups are represented by vertical lines and their unions are represented by horizontal lines at the level of the scale corresponding to the union between them.

Figure 2 – Illustrative data set.

Source: Elaborated by the author.

Some implementations of the hierarchical algorithms allow the user to specify, by means of a m_{CLSize} parameter, which is the minimum size of clusters that should be considered relevant. Thus, any cluster that is smaller than m_{CLSize} is automatically discarded as a spurious component or noise (CAMPELLO *et al.*, 2013). In a traditional hierarchy, all objects are labeled, even when they correspond to singletons. In addition, in a top-down interpretation of traditional hierarchies, a single object disconnecting itself from the other objects of a cluster is already characterized as a division in two clusters. In the hierarchies produced with $m_{CLSize} > 1$, all components smaller than m_{CLSize} are discarded and their objects are labeled 0, indicating that they do not belong to any cluster at that level of the hierarchy. These components are not interpreted as clusters and, therefore their appearance as they go through the hierarchy from top to bottom is not seen as a division of the original cluster to which they belonged.

Regardless using m_{CLSize} and the corresponding simplification of the hierarchy, in order to obtain a flat partition from a dendrogram or simplified hierarchy, it is necessary to choose a level of the scale and then make a horizontal cut. However, this methodology has limitations, such as the impossibility of selecting clusters at different hierarchical levels. These problems will be discussed later in [section 2.4](#).

Figure 3 – Dendrogram obtained from the illustrative data set (Figure 2), using the Euclidean Distance and the Single Linkage clustering.

Source: Elaborated by the author.

2.3 Density-based clustering

Density-based clustering algorithms can automatically find the number of clusters that exist in a data set. They also do not make assumptions about the underlying density function or the variance within the clusters (KRIEGEL *et al.*, 2011). As an effect, these algorithms do not necessarily group points with a low pairwise within-cluster dissimilarity, as measured by a dissimilarity function. The clusters do not necessarily have a convex shape but can be arbitrarily shaped in the data space. In the next section, we review basic concepts about popular density-based clustering algorithms and discuss the application of this type of algorithm on data analysis.

2.3.1 DBSCAN*, DBSCAN, and OPTICS

A number of concepts used later in this work refer back to ideas and definitions from DBSCAN (ESTER *et al.*, 1996), OPTICS (ANKERST *et al.*, 1999), and related algorithms. The key idea of these algorithms is that the density can be estimated for a particular point ($\mathbf{x}_p$) by counting the number of points (m_{pts}) within a specified radius (ϵ) of that point, including the point itself (TAN; QIN; LI, 2008). If there are at least m_{pts} objects belonging to ϵ -neighborhood of $\mathbf{x}_i$, then $\mathbf{x}_i$ is considered a **core point**. This concept is illustrated in Figure 4. In this case, the number of points inside the radius ϵ of point $\mathbf{x}$ is 4, including $\mathbf{x}$ itself.

Figure 4 – Core point.

Source: Elaborated by the author.

We start describing DBSCAN*, which is a simplified version of DBSCAN defined in terms of *core* and *noise objects* only (CAMPELLO; MOULAVI; SANDER, 2013):

Definition 1 (Core and Noise). An object $\mathbf{x}$ is called a core object w.r.t. ϵ and m_{pts} if its ϵ -neighborhood contains at least m_{pts} many objects, i.e., if $|N_\epsilon(\mathbf{x})| \geq m_{\text{pts}}$, where $|N_\epsilon(\mathbf{x})| = \{\mathbf{x}_i \in \mathbf{X} \mid d(\mathbf{x}, \mathbf{x}_i) \leq \epsilon\}$. An object is called noise if it is not a core object.

Definition 2 (ϵ -reachable). Two core objects $\mathbf{x}_i$ and $\mathbf{x}_j$ are ϵ -reachable w.r.t. ϵ and m_{pts} if $\mathbf{x}_i \in N_\epsilon(\mathbf{x}_j)$ and $\mathbf{x}_j \in N_\epsilon(\mathbf{x}_i)$.

Definition 3 (Density-connected). Two core objects $\mathbf{x}_i$ and $\mathbf{x}_j$ are density-connected w.r.t. ϵ and m_{pts} if they are directly or transitively ϵ -reachable.

Definition 4 (Cluster). A cluster $\mathbf{C}$ w.r.t. ϵ and m_{pts} is a non-empty maximal subset of $\mathbf{X}$ such that every pair of objects in $\mathbf{C}$ is density-connected.

As in DBSCAN, two parameters define a density threshold given by a minimum number of objects, m_{pts} , within a ball of radius ϵ centered at an object $\mathbf{x}$. Clusters are formed only by objects $\mathbf{x}$ satisfying this minimum density threshold (core objects). Two such objects are in the same cluster if and only if they can reach one another directly or through a chain of objects in which every consecutive pair is within each other's ϵ -neighborhood.

DBSCAN* is not only simpler, but it is also statistically more rigorous than the original DBSCAN, as it strictly conforms with the classic principle of *density-contour clusters* as defined by (HARTIGAN, 1975). The original DBSCAN relaxes this principle by allowing some objects below the density threshold, called *border objects*, to be incorporated into clusters. Specifically, a border object in DBSCAN is a non-core object that lies within the ϵ -neighborhood of a core object. For the sake of convenience, here we will formalize this notion by using the following definitions adapted from OPTICS (ANKERST *et al.*, 1999):

Definition 5 (Core Distance). The core distance of an object $\mathbf{x}_i \in \mathbf{X}$ w.r.t. m_{pts} , $d_{\text{core}}(\mathbf{x}_i)$, is the distance from $\mathbf{x}_i$ to its m_{pts} -nearest neighbor (including $\mathbf{x}_i$).

Definition 6 (Reachability Distance). The (asymmetric) reachability distance from an initial object $\mathbf{x}_i$ to an end object $\mathbf{x}_e$ w.r.t. m_{pts} , $d_{\text{reach}}(\mathbf{x}_i, \mathbf{x}_e)$, is the largest of the core distance of $\mathbf{x}_i$ and the distance between $\mathbf{x}_i$ and $\mathbf{x}_e$: $d_{\text{reach}}(\mathbf{x}_i, \mathbf{x}_e) = \max\{d_{\text{core}}(\mathbf{x}_i), d(\mathbf{x}_i, \mathbf{x}_e)\}$.

From Definition 5, it is clear that the core distance of an object $\mathbf{x} \in \mathbf{X}$ is the minimum value of the radius ε for which $\mathbf{x}$ is a core object (i.e., its density is above the minimum density threshold). By definition, a border object $\mathbf{x}_e$ in DBSCAN is not a core object, which means $\varepsilon < d_{\text{core}}(\mathbf{x}_e)$. Also by definition, a border object $\mathbf{x}_e$ falls within the ε -neighborhood of a core object, say $\mathbf{x}_i$, which means $d(\mathbf{x}_i, \mathbf{x}_e) \leq \varepsilon$ and, since $\mathbf{x}_i$ is core, $\varepsilon \geq d_{\text{core}}(\mathbf{x}_i)$. From these inequalities, it follows that $d_{\text{core}}(\mathbf{x}_e) > \varepsilon \geq \max\{d_{\text{core}}(\mathbf{x}_i), d(\mathbf{x}_i, \mathbf{x}_e)\}$, and using Definition 6 a border object can then be defined as:

Definition 7 (Border Object). An object $\mathbf{x}_e \in \mathbf{X}$ is called a border object w.r.t. ε and m_{pts} if $d_{\text{core}}(\mathbf{x}_e) > \varepsilon$ and there exists another object $\mathbf{x}_i \in \mathbf{X}$ such that $\varepsilon \geq d_{\text{reach}}(\mathbf{x}_i, \mathbf{x}_e)$.

The clusters in the original DBSCAN are the same as in DBSCAN*, augmented with their corresponding border objects; all the other objects are labeled as noise by both algorithms. From a graph perspective, it is straightforward to see that the clusters in DBSCAN* (Definition 4) are the connected components of an undirected graph where each core object is represented as a vertex and two vertices are adjacent if and only if the corresponding core objects fall within each other's ε -neighborhood (i.e., iff they are ε -reachable — Definition 2). DBSCAN also includes border objects as vertices, each of which is adjacent to a core object. For a border object $\mathbf{x}_e$, if there is more than one core object $\mathbf{x}_i$ satisfying Definition 7, the original DBSCAN makes $\mathbf{x}_e$ adjacent to one of those chosen randomly, but the choice can be made deterministically, e.g., the one that minimizes $d_{\text{reach}}(\mathbf{x}_i, \mathbf{x}_e)$ (or $d(\mathbf{x}_i, \mathbf{x}_e)$ in case of ties).

Starting from an arbitrary object in the dataset, OPTICS (ANKERST *et al.*, 1999) derives an *ordering* ($\prec$) of the data objects that implicitly encodes all possible DBSCAN solutions for a given value of m_{pts} . The algorithm does not require the radius ε to produce such an ordering, which has the following property: given an object $\mathbf{x}_q$, the smallest reachability distance to $\mathbf{x}_q$ from any of its preceding objects is no greater than the smallest reachability distance from any of its preceding objects to an object succeeding $\mathbf{x}_q$, i.e.,

$$\min_{\mathbf{x}_p: \mathbf{x}_p \prec \mathbf{x}_q} d_{\text{reach}}(\mathbf{x}_p, \mathbf{x}_q) \leq \min_{\substack{\mathbf{x}_o, \mathbf{x}_r: \\ \mathbf{x}_o \prec \mathbf{x}_q \prec \mathbf{x}_r}} d_{\text{reach}}(\mathbf{x}_o, \mathbf{x}_r).$$

This property is important for two reasons: (a) it ensures that by plotting $\min_{\mathbf{x}_p: \mathbf{x}_p \prec \mathbf{x}_q} d_{\text{reach}}(\mathbf{x}_p, \mathbf{x}_q)$ for every object $\mathbf{x}_q$ in the given order, the so-called OPTICS *reachability plot*, presented in Figure 5, where density-based clusters and sub-clusters appear as valleys or “dents” in the plot; and (b) if one wants to set a threshold ε , as a horizontal line cutting through the plot, it is straightforward to show that DBSCAN clusters with radius ε correspond essentially to the contiguous subsequences of the ordered points for which the plot is below the threshold.

Figure 5 – Reachability-plot for a two dimensional data set. Valleys in the reachability plot may indicate the presence of clusters on the data.

Source: Elaborated by the author.

The OPTICS ordering and reachability plot can be easily achieved by keeping an adaptable priority queue sorted by the smallest reachability distance from an object outside the queue (already processed) to each object inside the queue. At each iteration, the object with the smallest such distance (priority key) is removed from the queue (processed), the reachability distances from that object to the objects inside the queue are computed, and the queue is readjusted accordingly. From a graph perspective, this is algorithmically analogous to Prim's algorithm to compute a Minimum Spanning Tree (MST). The only difference is that OPTICS operates on a directed graph where each pair of vertices $(\mathbf{x}_p, \mathbf{x}_q)$ is connected by a pair of unique edges, one in each direction, whose weights are the corresponding reachability distances, i.e. $d_{\text{reach}}(\mathbf{x}_p, \mathbf{x}_q)$ and $d_{\text{reach}}(\mathbf{x}_q, \mathbf{x}_p)$. The optional threshold ε to extract DBSCAN clusters corresponds to pruning out from such a complete digraph any edge whose weight is larger than ε . The strongly connected components of the resulting digraph (subsets of vertices mutually reachable via directed paths) correspond to the DBSCAN* clusters with radius ε , whereas the DBSCAN clusters additionally include vertices that are not part of any of the strongly connected components, but are reachable from those (i.e., the border objects).

2.3.2 HDBSCAN*

Proposed by [Campello, Moulavi and Sander \(2013\)](#), the Hierarchical DBSCAN* (HDBSCAN*) is considered as the state of the art for the density-based hierarchical clustering algorithms. In [Campello et al. \(2015\)](#), it is extended to other domains, such as semi-supervised clustering, becoming a framework that can perform: (i) unsupervised clustering; (ii) semi-supervised

clustering, where *should-link* and *should-not-link* constraints are provided; (iii) outlier detection; and (iv) data visualization.

Given any value for m_{pts} , the HDBSCAN* is able to efficiently calculate (possibly, with the same speed of a single DBSCAN run) a partition hierarchy for all possible values of $\varepsilon \in [0, \infty)$. Each hierarchical level (each partition) is associated with a value of ε which, if used by DBSCAN*, would produce exactly the same solution. If a DBSCAN solution is desired, it is sufficient to cut the hierarchy at the corresponding ε level and add the border objects (which can be done in linear time).

HDBSCAN* is also able to create a postprocessed hierarchy called a *simplified hierarchy* that describes a cluster tree containing only the most important clusters. This simplified hierarchy can be represented in a compact way, therefore it is necessary to store only the hierarchical levels where the actual division of a cluster occurred, and all levels are discarded in which a cluster only “shrinks” by losing objects that are no longer considered dense becoming noise (or by loss of spurious components, if the optional use of a minimum cluster size, m_{CISize} , is adopted).

To identify all possible density-based clusters in the interval $\varepsilon \in [0, \infty)$ HDBSCAN* applies the Definitions described in subsection 2.3.1, except for the asymmetric version of the reachability distance and border objects (Definitions 6 and 7, respectively). HDBSCAN* uses one additional concept, the mutual reachability distance, defined as:

Definition 8 (Mutual Reachability Distance). The mutual reachability distance between two objects $\mathbf{x}_p$ and $\mathbf{x}_q$ in $\mathbf{X}$ w.r.t. m_{pts} is defined as $d_{mreach}(\mathbf{x}_p, \mathbf{x}_q) = \max\{d_{core}(\mathbf{x}_p), d_{core}(\mathbf{x}_q), d(\mathbf{x}_p, \mathbf{x}_q)\}$.

Figure 6 – Mutual reachability distance for objects $\mathbf{x}_p$ and $\mathbf{x}_q$.

Source: Elaborated by the author.

The mutual reachability distance (Definition 8) is the minimum radius ε such that $\mathbf{x}_p$ and $\mathbf{x}_q$ are ε -reachable according to Definition 2. Figure 6 shows an example of the core distance and the mutual reachability distance (for $m_{pts} = 4$). For a given m_{pts} , ε sets a density threshold $\frac{1}{\varepsilon}$, inversely proportional to its value. Therefore, the mutual reachability distance is inversely proportional to the highest density threshold for which $\mathbf{x}_p$ and $\mathbf{x}_q$ are connected (Definition 3).

HDBSCAN* provides as a result a complete hierarchy composed of all possible solutions that could be found by DBSCAN*, given any value of m_{pts} and an infinite range of values for $\varepsilon \in [0, \infty)$. To do this, the following graph of transformed distances is used (conceptually) (CAMPELLO; MOULAVI; SANDER, 2013; CAMPELLO *et al.*, 2015):

Definition 9 (Mutual Reachability Graph). The mutual reachability graph is a complete graph, $\mathbb{G}_{m_{pts}}$, in which the objects of $\mathbf{X}$ are vertices and the weight of each edge is the mutual reachability distance (w.r.t. m_{pts}) between the respective pair of objects.

Let $\mathbb{G}_{m_{pts}, \varepsilon} \subseteq \mathbb{G}_{m_{pts}}$ be the subgraph obtained by removing all the edges that have a weight greater than a given value of ε . From the arguments presented above, it can be seen that the clusters identified by DBSCAN*, with respect to m_{pts} and ε , correspond to the related components of $\mathbb{G}_{m_{pts}}$, and the objects labeled noise to the isolated vertices by removing these edges. This relationship allows all clusters that DBSCAN* is able to find ($\varepsilon \in [0, \infty)$) to be generated in a hierarchical way by removing the edges of $\mathbb{G}_{m_{pts}}$ in descending order of weight. It is interesting to observe that this is essentially the graph-based definition of the Single-Linkage (JAIN; DUBES, 1988) hierarchical clustering algorithm, and therefore there is a conceptual relationship between the DBSCAN* and SL algorithms on the transformed space of the mutual reachability distances. The clustering obtained by DBSCAN* with respect to a m_{pts} and some value of ε is identical to first executing the algorithm SL on the transformed space of mutual reachability distances, then cutting the resulting dendrogram at the ε level, and treating all singletons with $d_{core} > \varepsilon$ as noise. This shows that it is possible to create a hierarchical version of DBSCAN* by applying an algorithm that computes a SL hierarchy over the “transformed space” of mutual reachability distances.

One of the most efficient ways to calculate a SL hierarchy is by using a divisive algorithm that acts by removing the edges, in descending order of weight, from a minimal spanning tree (MST) (JAIN; DUBES, 1988). In the present context, it is an MST in the transformed space of mutual reachability distances, called MST_r . In order to represent the level of the hierarchy from which an object becomes a noise ($d_{core} > \varepsilon$) and above which it can be part of a cluster, even a cluster by itself (i.e., a singleton), HDBSCAN* extends MST_r with the addition of loops, with a weight equal to the core distance (d_{core}) of the respective object represented by the vertex. Thus, in general, HDBSCAN*: (i) computes MST_r using some computationally efficient method (i.e., Prim’s algorithm); (ii) adds to MST_r self-edges containing the core distance of each vertex; and (iii) removes the edges in decreasing order (in case of a tie, all the edges are removed simultaneously) and labels the connected components as clusters.

HDBSCAN* also optionally allows a minimum size m_{CSize} for the clusters produced (commonly used $m_{CSize} = m_{pts} = 4$). Any component that is less than the m_{CSize} parameter is not interpreted as a cluster and is automatically dropped from the hierarchy as noise. In Table 1, the hierarchy produced by HDBSCAN* for the data set of Figure 2 is shown.

Table 1 – Cluster hierarchy with $m_{pts} = m_{CLSize} = 3$ obtained from the illustrative dataset presented in Figure 2. The first line contains the labels of all the objects in the dataset, and the column ε presents the density level where one object becomes part of a cluster or a noise.

ε	X ₁	X ₅	X ₂₀	X ₂	X ₃	X ₁₃	X ₁₄	X ₁₂	X ₁₅	X ₁₆	X ₄	X ₂₂	X ₁₇	X ₁₈	X ₁₉	X ₂₁	X ₆	X ₉	X ₁₁	X ₇	X ₈	X ₁₀
3.30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3.19	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
3.18	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.24	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.22	0	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.17	0	0	0	0	4	4	4	4	4	4	5	5	5	5	5	5	3	3	3	3	3	3
1.00	0	0	0	0	4	4	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.98	0	0	0	0	0	4	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.85	0	0	0	0	0	0	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.82	0	0	0	0	0	0	4	4	4	4	0	0	5	5	5	5	3	3	3	3	3	3
0.72	0	0	0	0	0	0	0	4	4	4	0	0	5	5	5	5	3	3	3	3	3	3
0.63	0	0	0	0	0	0	0	4	4	4	0	0	0	5	5	5	3	3	3	3	3	3
0.61	0	0	0	0	0	0	0	4	4	4	0	0	0	5	5	5	6	6	6	7	7	7
0.60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6	6	7	7	7
0.51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6	6	0	0	0
0.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Source: Elaborated by the author.

Table 2 – Compact hierarchy with $m_{pts} = m_{CLSize} = 3$ obtained from the illustrative dataset presented in Figure 2.

ε	X ₁	X ₅	X ₂₀	X ₂	X ₃	X ₁₃	X ₁₄	X ₁₂	X ₁₅	X ₁₆	X ₄	X ₂₂	X ₁₇	X ₁₈	X ₁₉	X ₂₁	X ₆	X ₉	X ₁₁	X ₇	X ₈	X ₁₀
3.30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3.19	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.17	0	0	0	0	4	4	4	4	4	4	5	5	5	5	5	5	3	3	3	3	3	3
0.61	0	0	0	0	0	0	0	4	4	4	0	0	0	5	5	5	6	6	6	7	7	7
0.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Source: Elaborated by the author.

Full hierarchies such as Table 1 can be represented in a compact way in which only the consecutive pairs of hierarchical levels where a real division of a cluster occurs in two or more clusters are stored, i.e. all levels are discarded in which only objects are no longer dense or spurious components arise and are treated as noise, without dividing a cluster into new clusters at this level. Table 2 presents the compressed hierarchy of Figure 2.

Finally, in addition to a complete or compressed hierarchy, HDBSCAN* also generates a cluster tree, as shown in Figure 7, formed only by the most relevant clusters, and optionally as post-processing, it can extract a flat partition from a quality measure for clusters. In section 2.4, the strategy of extracting clusters used by HDBSCAN*, called FOSC, is discussed.

Figure 7 – Cluster tree generated from the hierarchy shown in Table 2.

Source: Elaborated by the author.

2.3.2.1 HDBSCAN* with all-points core distance

Campello, Moulavi and Sander (2013), Campello *et al.* (2015) showed that the parameter m_{pts} , which is commonly shared by all density-based algorithms previously discussed, corresponds to a classic smoothing factor of a nonparametric, nearest neighbor density estimate. This parameter is not critical and can be useful for providing the user with fine-tuning control of the results, e.g., by visual inspection of the clustering hierarchies or of the reachability plots. It can be removed though, if desired, basically by replacing the core and mutual reachability distances in Definitions 5 and 8 with a parameterless alternative. In particular, a parameterless version of HDBSCAN* was proposed (MOULAVI, 2014) that is based on a new core distance of an object, which does not depend on its m_{pts} -neighborhood, but rather considers the dataset in a way that closer objects contribute more to the density than farther objects do:

Definition 10 (All-points core distance). The all-points core-distance of a d -dimensional point $\mathbf{x}$ of a dataset $\mathbf{X}$ with respect to all other $n - 1$ points in $\mathbf{X}$, i.e., $\mathbf{X} \setminus \{\mathbf{x}\}$, is defined as (MOULAVI, 2014):

$$d_{\text{aptsCore}}(\mathbf{x}) = \left(\frac{\sum_{\mathbf{x}_i \in \mathbf{X} \setminus \{\mathbf{x}\}} \left(\frac{1}{d(\mathbf{x}, \mathbf{x}_i)} \right)^d}{N - 1} \right)^{-\frac{1}{d}}$$

Let us note that the all-points core distance is only meaningful for datasets as points in a d -dimensional real vector space, as opposed to the original HDBSCAN* (as well as DBSCAN, OPTICS) that can operate with any dataset for which some type of pairwise distance between objects can be defined.

2.4 FOSC

HDBSCAN* offers an optional post-processing method of its clustering hierarchy, called FOSC, which can extract a flat clustering solution by performing local cuts through the hierarchy in order to select a collection of non-overlapping clusters (and, possibly, objects unclustered as noise) that is optimal according to a given unsupervised or semi-supervised criterion. FOSC is unique in that it can perform non-horizontal cuts through a hierarchy, which means that clusters can be extracted from different hierarchical levels. In HDBSCAN*, this means that solutions composed of clusters at various density levels can be obtained, which could not be obtained by a conventional, global horizontal cut at a single hierarchical level.

The Framework for Optimal Extraction of Clusters (FOSC) was proposed by [Campello et al. \(2013\)](#) as a general framework to perform optimal extraction of flat clustering solutions from clustering hierarchies. In order to understand how the method operates in HDBSCAN*, let us consider an example. [Figure 2](#) shows a toy dataset with 22 objects. The complete clustering hierarchy produced by HDBSCAN* with $m_{pts} = 3$ is shown in [Table 1](#), where rows correspond to hierarchical levels (density thresholds for varied ϵ), columns correspond to data objects, and entries contain cluster labels (“0” stands for noise). Notice that, unlike traditional dendrograms, clusters can shrink and yet retain the same label when individual objects (or spurious components with fewer than an optional, user-defined minimum cluster size, m_{CSize}) are disconnected from them becoming noise, as the density threshold increases for decreasing values of ϵ (top-down the hierarchy). Only when a cluster is divided into two non-spurious subsets of density-connected objects the resulting subsets are deemed new clusters. Thus, the complete hierarchy in [Table 1](#) can actually be represented as a simplified cluster tree where the root (C_1) is the “cluster” containing the whole dataset, which subdivides into two child nodes corresponding to clusters C_2 and C_3 , and these further subdivide into two sub-clusters each (C_4 and C_5 from C_2 , C_6 and C_7 from C_3).

Notice in [Table 1](#) that objects belonging to a parent cluster do not necessarily belong to any of its children, as they may become noise before a cluster splits. For instance, objects x_1 , x_2 , x_5 , and x_{20} belong to C_2 but not to C_4 or C_5 . Technically, each of these objects is assigned an individual node on its own in the cluster tree, in this example all as descendants from C_2 ¹. As we will see later, these singleton nodes can only affect cluster extraction in the semi-supervised scenario, and only if the corresponding objects are pre-labeled.

[Figure 8](#) illustrates the cluster tree corresponding to the clustering hierarchy in [Table 1](#), along with all the information needed to run FOSC in the unsupervised setting. Singleton nodes corresponding to noise objects are omitted for the sake of clarity, as they do not affect computations.

If clusters in the cluster tree can be properly assessed according to a suitable measure of

¹ There is no such a thing as a “rag bag cluster” because noise objects should not be seen as clustered with each other, in spite of sharing a common label “0”.

Figure 8 – Cluster tree and unsupervised FOOSC extraction with stability.

Source: Elaborated by the author.

cluster quality², an optimal flat solution in which objects are guaranteed not to belong to more than one cluster can be extracted by FOOSC. Formally, let $\{C_1, \dots, C_k\}$ be the set of all candidate clusters in the cluster tree from which we want to extract a flat solution, $\mathbf{P}$. Assume that there is an objective function $J_T(\mathbf{P})$ that we want to maximize, such that J_T can quantitatively assess the quality of every valid candidate solution $\mathbf{P}$. Functional $J_T(\mathbf{P})$ must be decomposable according to two properties:³

1. **Additivity:** $J_T(\mathbf{P})$ must be written as the sum of individual components $J(C_i)$, each of which is associated with a single cluster C_i of $\mathbf{P}$;
2. **Locality:** Every component $J(C_i)$ must be computable locally to C_i , regardless of what the other clusters that compose the candidate solution $\mathbf{P}$ are.

Due to the property of *locality*, the value $J(C_i)$ associated with every cluster in the cluster tree can be computed beforehand, i.e., prior to the decision in which clusters will compose the final solution to be extracted. These are the values illustrated below each node in Figure 8.

Due to the property of *additivity*, the objective function can be written as $J_T(\mathbf{P}) = \sum_{C_i \in \mathbf{P}} J(C_i)$, and the problem we want to solve is to choose a collection $\mathbf{P}$ of clusters such that: (a) $J_T(\mathbf{P})$ is maximized; and (b) $\mathbf{P}$ is a valid flat solution, i.e., clusters and their sub-clusters are mutually exclusive (no data object belongs to more than one cluster). Mathematically, the

² Such measures can be unsupervised or semi-supervised.

³ Both properties could be seen as related to the “locality” property of *clustering functions* as introduced by Ackerman, Ben-David and Loker (2010).

optimization problem can be formulated as (CAMPELLO *et al.*, 2013):

$$\begin{aligned}
& \max_{\delta_1, \dots, \delta_k} \quad \sum_{i=1}^k \delta_i J(\mathbf{C}_i) \\
& \text{s. t.} \quad \delta_i \in \{0, 1\}, \quad i = 1, \dots, k. \\
& \quad \quad \sum_{j \in \mathbf{I}_h} \delta_j = 1, \quad \forall h \text{ such that } \mathbf{C}_h \text{ is a leaf node/cluster.}
\end{aligned} \tag{2.2}$$

where δ_i ($i = 1, \dots, k$) is an indicator function that denotes whether cluster $\mathbf{C}_i$ is selected to be part of the solution ($\delta_i = 1$) or not ($\delta_i = 0$), and $\mathbf{I}_h$ is the set of cluster indices on the path from any external node $\mathbf{C}_h$ up to the root $\mathbf{C}_1$. Note that the constraints ensure that a single cluster is selected in any branch from the root to a leaf.

FOSC solves this problem taking advantage of the fact that, due to the locality property of the cluster quality measure J , the partial selections made inside any subtree remain optimal in the context of larger trees containing that subtree. This allows for a very efficient, globally optimal dynamic programming method that traverses the cluster tree bottom-up starting from the leaves, comparing the quality of parent clusters against the aggregated quality of the respective subtrees, carrying the optimal choices upwards until the root is reached.

In Figure 8, notice that the sum of $J(\mathbf{C}_4) = 3.23$ and $J(\mathbf{C}_5) = 3.41$ (as well as the hidden singleton nodes descending from $\mathbf{C}_2$, all of which have J value of zero) is equal to 6.64, which is smaller than $J(\mathbf{C}_2) = 7.28$, hence $\mathbf{C}_2$ is temporarily selected while its subtrees are discarded. Analogously, the aggregated value of $J(\mathbf{C}_6)$ and $J(\mathbf{C}_7)$ (1.4) is compared against $J(\mathbf{C}_3) = 7.67$, which is larger, hence $\mathbf{C}_3$ is temporarily selected whereas $\mathbf{C}_6$ and $\mathbf{C}_7$ are discarded. Now, the sum of $J(\mathbf{C}_2)$ and $J(\mathbf{C}_3)$ (14.95) is larger than $J(\mathbf{C}_1) = 6.67$, hence $\mathbf{C}_1$ is discarded while $\mathbf{C}_2$ and $\mathbf{C}_3$ are retained. Since the root has been reached, the final solution is $\mathbf{P} = \{\mathbf{C}_2, \mathbf{C}_3\}$, with $J_T(\mathbf{P}) = 14.95$.

While many clustering quality criteria from the literature satisfy the additive property, it is not easy to find criteria that satisfy the locality property required by FOSC. In the original publication (CAMPELLO *et al.*, 2013), a criterion was introduced that is based on the classic notion of *cluster lifetime*. The lifetime of a cluster in a clustering dendrogram is basically the length of the dendrogram scale along which the cluster exists (JAIN; DUBES, 1988). More prominent clusters persist longer across multiple hierarchical levels, and therefore they have a longer lifetime.

This concept has been adapted by Campello *et al.* (2013) to account for the fact that in certain hierarchies, including density-based hierarchies such as the one in Table 1, not all data objects stay in the cluster during its whole lifetime, because some objects become noise along the way. In other words, objects have different lifetimes as part of a cluster. The unsupervised measure of *Stability* of a cluster as proposed by Campello *et al.* (2013) is the sum of the lifetimes of every object in that cluster, $J(\mathbf{C}_i) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \text{lifetime}(\mathbf{x}_j)$.

For example, in [Table 1](#), cluster C_4 appears bottom-up at level 0.61 (formed by objects x_{12} , x_{15} , and x_{16}) and disappears when it is merged with cluster C_5 , giving rise to C_2 at level 1.22. Along this interval, another three objects join this cluster, at levels 0.82 (x_{14}), 0.98 (x_{13}), and 1.00 (x_3). Hence, its Stability is given by $J(C_4) = 3 * (1.22 - 0.61) + (1.22 - 0.82) + (1.22 - 0.98) + (1.22 - 1.00) = 2.69$.

To perform flat cluster extraction in an *unsupervised* way, HDBSCAN* uses FOSC with the Stability criterion as described above, except that it replaces ε with $\frac{1}{\varepsilon}$ in the scale (which is therefore flipped) for the computation of lifetime. This makes Stability more statistically sound in the density-based context as it becomes equivalent to the concept of relative *excess of mass* of a cluster ([CAMPELLO et al., 2015](#)).⁴ In this case, the Stability of cluster C_4 in our example above would be computed as $J(C_4) = 3 * (1/0.61 - 1/1.22) + (1/0.82 - 1/1.22) + (1/0.98 - 1/1.22) + (1/1.00 - 1/1.22) = 3.23$. These are precisely the values shown below each cluster in [Figure 8](#).

The Stability of any singleton node containing a noise object is defined as zero as the lifetime of noise is undefined. Thus, singleton nodes with noise do not directly affect unsupervised cluster extraction. Indirectly though, larger amounts of noise in a final solution $\mathbf{P}$ are indirectly penalized as noise objects and do not add anything to the Overall Stability of that solution.

FOSC can operate in a semi-supervised way if a suitable measure that takes into account semi-supervision is provided. However, the existing method, currently used by HDBSCAN*, is based on the maximization of the number of (soft, as preferences only) should-link and should-not-link constraints that are satisfied in the extracted clusters ([CAMPELLO et al., 2013](#); [CAMPELLO et al., 2015](#)). Obviously, given a collection of labeled objects, one can produce constraints by creating should-link relations between pairs of objects with the same label, and should-not-link relations between pairs of objects with different labels. Nevertheless, there are two main disadvantages in working with pairwise constraints, rather than directly with labels. The first one is the additional effort to generate the constraints, which is actually unnecessary as we will discuss later ([Chapter 6](#)). Second, as the number of different labels increase, so does the imbalance between the number of should-not-link constraints and the number of should-link constraints that follow on from the labels, as the number of should-not-link constraints increases much more with adding additional labels than the number of should-link constraints. Thus, the maximization of the number of constraints satisfied tends to be biased towards satisfying should-not-link relations, which may produce unexpected results. As a matter of fact, it has been observed in the semi-supervised clustering literature that adding constraints may possibly decrease the performance of clustering algorithms ([DAVIDSON; WAGSTAFF; BASU, 2006](#); [CRAENENDONCK; BLOCKEEL, 2017](#)).

⁴ It also makes it possible to compute Stability for the root, C_1 .

2.5 Final remarks

In this chapter, we described the main concepts of cluster analysis, presenting the basic characteristics, such as partitional and hierarchical algorithms, to some more advanced concepts which are density-based clustering and optimal cluster extraction from cluster hierarchies. Density-based clustering and optimal extraction from cluster hierarchies were fundamental in understanding and developing the contributions in this thesis. In the next chapter, we present concepts related to semi-supervised learning (semi-supervised classification and semi-supervised clustering).

SEMI-SUPERVISED LEARNING

Semi-supervised learning algorithms implement circumstances where a small amount of labeled data and a large amount of unlabeled data is available for training (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; ZHU; GOLDBERG, 2009).

Typically, semi-supervised learning algorithms are based on extensions of either supervised or unsupervised algorithms by including additional information in the form originally handled by the other learning paradigm. As example, in semi-supervised *clustering*, a collection of labeled observations can be used to guide the (otherwise unsupervised) search for clustering solutions that better meet users' prior expectations. Labels in clustering only indicate whether observations are expected to be part of the same cluster or different clusters, there is no one-to-one association between the unique labels known to the user and the possible categories to be discovered in the data. In semi-supervised *classification*, the classes are known in advance, and unlabeled observations are used in addition to the labeled ones to improve on (the otherwise supervised) classification performance.

Semi-supervised learning can be categorized into *inductive* and *transductive* learning. Inductive learning uses both labeled and unlabeled training data to generate a model able to predict the labels for the unlabeled training data as well as for future data to be labeled. Transductive learning predicts the labels of the unlabeled training data only (from which a model may *optionally* be derived afterwards, if prediction of new, unseen data objects is required). In transductive *classification*, a large amount of unlabeled objects can be classified based on a small fraction of labeled objects, which is not representative enough to successfully train a classifier in a traditional, fully supervised way. Label-based semi-supervised *clustering*, is intrinsically transductive in its nature. The main difference from transductive classification is that in clustering not necessarily all possible categories and their labels are known in advance, so unlabeled objects may be assigned newly discovered labels that are not present in the original training set.

In the case of semi-supervised classification, unlabeled data can help improve classification performance depending on a good matching of problem structure with *model assumption*: “...there’s no free lunch. Bad matching of problem structure with model assumption can lead to degradation in classifier performance” (ZHU, 2005). Different models of semi-supervised classification exist, relying on different model assumptions (ZHU, 2005). One of the major paradigms, clustering-based models, follows the well-known *cluster assumption* of semi-supervised classification:

Assumption 1 (Cluster Assumption (CHAPELLE; SCHÖLKOPF; ZIEN, 2006)). If points are in the same cluster, they are likely to be of the same class.

This assumption is quite general and broad in scope as there are many possible interpretations of “cluster”, under different clustering paradigms. An important, statistically sound paradigm is *density-based clustering* (KRIEGEL *et al.*, 2011), described in Chapter 2, where clusters are defined as high-density data regions separated by low-density regions. Under this paradigm, Assumption 1 is closely related to another common assumption in semi-supervised classification:

Assumption 2 (Smoothness Assumption (CHAPELLE; SCHÖLKOPF; ZIEN, 2006)). The label function is smoother in high-density than in low-density regions... If two points in a high-density region are close, then so should be their outputs (labels)... If, on the other hand, they are separated by a low-density region, then their outputs need not be close.

According to Chapelle, Schölkopf and Zien (2006), in the context of classification and from a density-based clustering perspective, Assumptions 1 and 2 are equivalent to each other and can be read as “*The decision boundary should lie in a low-density region (low density separation)*”. In spite of the obvious connections between these two areas, however, the use of density-based clustering for semi-supervised classification has been surprisingly overlooked in the literature. Many methods focus instead on the use of graphs (as opposed to clusters) to model the notions of locality and connectivity of the data (SOUSA; REZENDE; BATISTA, 2013; SOUSA, 2015). Such graph-based methods mostly rely on the following model assumption:

Assumption 3 (Graph Assumption (ZHU; GOLDBERG, 2009)). Class labels are “smooth” with respect to the graph, so that they vary slowly, i.e., if two points are connected by a strong edge, their labels tend to be the same.

In the context of semi-supervised classification, different categories of algorithms have been described in the literature (ZHU, 2005). Closer to our proposed work, described in Chapter 5, are the *clustering-based* and the *graph-based* approaches. Graph-based algorithms construct a graph with vertices from both labeled and unlabeled objects. Generally, neighboring vertices are connected by edges such that edge weights are proportional to some measure of local

connectivity strength (SOUSA; REZENDE; BATISTA, 2013). Once the neighborhood graph is built, labels can be somehow transferred from labeled to unlabeled objects, e.g., using Markov chain propagation techniques (SZUMMER; JAAKKOLA, 2002) or regularized methods based on the graph Laplacian (ZHAO *et al.*, 2006). The Laplacian SVM (LapSVM) (BELKIN; NIYOGI; SINDHWANI, 2006) is related to the latter category and is a state-of-the-art algorithm in the semi-supervised classification literature.

As an example, observe Figure 9 that was build with both unlabeled and labeled objects. The last one are represented by objects x_1 (red) and x_2 (green). From a graph-based perspective, there is a “stronger” connection between objects x_1 and x_3 , even the object x_3 is closer in distance to x_2 (e.g, Euclidean distance). In this scenario, x_3 has a high probability of obtain the label from x_1 (red label).

Figure 9 – A graph composed from both labeled (x_1 and x_2 and unlabeled objects. The unlabeled object x_3 will obtain the label from x_1 even though it is closer (i.e., in Euclidean distance) from x_2 .

Source: Elaborated by the author.

A well-known strategy for label propagation in graph-based semi-supervised classification is the use of a so-called harmonic function. In this context, a harmonic function is a function that has the same values as the labels on the labeled data, and satisfies the weighted average property on the unlabeled data, i.e., the value assigned to each unlabeled object is the weighted average of the values of its neighbors. In a binary classification problem, the class labels are coded, e.g., as $\{-1, +1\}$, and these values are assigned to the vertices corresponding to the labeled objects. Each remaining (unlabeled) object has its value determined as the average of the values of its adjacent vertices in the neighborhood graph, weighted by the corresponding edge weights. The resulting real values, which allow for different physical and probabilistic interpretations, can be discretized back into $\{-1, +1\}$ to achieve the final transductive classification. A classic algorithm that follows this type of approach is the Gaussian Field Harmonic Function (GFHF) (ZHU; GHAHRAMANI; LAFFERTY, 2003).

To address multi-class semi-supervised classification, [Liu and Chang \(2009\)](#) formulated a constrained label propagation problem by incorporating class priors, leading to a simple closed-form solution. The algorithm, called Robust Multi-Class Graph Transduction (RMGT), is an extension of the GFHF algorithm, which can be viewed as a constrained optimization problem using a graph Laplacian as smoothness measure ([SOUSA, 2015](#)). Both RMGT and GFHF, as well as the previously mentioned LapSVM algorithm, are used as baseline for comparisons in our experimental evaluation, described in [Chapter 5](#).

In contrast to graph-based methods, clustering-based algorithms for semi-supervised classification perform label transduction based on the clustering structure of the data, rather than by using an explicit graph ([ZHU; GOLDBERG, 2009](#)). However, since certain clustering techniques are implicitly or explicitly built upon graphs and related algorithms, there are connections between these two different paradigms of semi-supervised learning, which are investigated in this document. Of particular interest in our context are density-based clustering methods ([KRIEGEL et al., 2011](#)), which are popular in the data mining field as a statistically sound approach that has also been used for semi-supervised classification, and has also been shown to have strong connections with elements from graph theory ([CAMPELLO et al., 2015](#)). Two noticeable algorithms in this context are HISSCLU ([BÖHM; PLANT, 2008](#)) and Semi-Supervised DBSCAN (SSDBSCAN) ([LELIS; SANDER, 2009](#)), both of which are built upon notions inherited from two classic, widely used unsupervised density-based clustering algorithms, namely, OPTICS ([ANKERST et al., 1999](#)) and DBSCAN ([ESTER et al., 1996](#)).

SSDBSCAN ([LELIS; SANDER, 2009](#)), which has more recently also been extended to the active learning scenario ([LI et al., 2014](#)), was in principle proposed as a semi-supervised clustering method, which does not necessarily label all objects (as it would normally be expected in transductive classification), but rather leave certain objects unlabeled as noise, as usual (and meaningful) in density-based *clustering* applications. Despite this, the algorithm explicitly relies on a *classification* assumption:

Assumption 4 (Classification Assumption). There is at least one (possibly more) labeled object from each class.

Unlike SSDBSCAN, HISSCLU ([BÖHM; PLANT, 2008](#)) already includes an extended label-propagation scheme that assigns a label from the training set to every unlabeled object in the database. It further differs from SSDBSCAN in that it also includes a preprocessing mechanism to widen the gap between nearby classes by stretching distances between objects around class boundaries. This mechanism allows HISSCLU to expand different class labels even within clusters that are formed by more than one class (i.e., clusters of objects that are density connected but not pure in their labels). SSDBSCAN, in contrast, makes the *label consistency* assumption:

Assumption 5 (Label Consistency Assumption). “Label consistency requires different labels to belong to different clusters; under this assumption, a single class can still have multiple modes [clusters or sub-clusters]; in other words, objects in different clusters can have the same label, only in a single cluster the labels have to be the same”(LELIS; SANDER, 2009).

From this perspective, SSDBSCAN relies more strictly than HISSCLU on the *clustering assumption of semi-supervised classification* (Assumption 1). Unlike SSDBSCAN, which was originally proposed for the semi-supervised clustering task, HISSCLU can perform both semi-supervised classification and clustering. Both remain state-of-the-art algorithms in the density-based literature, so they are also used as baseline for comparisons in our experimental evaluation (Chapters 5 and 6).

Apart from SSDBSCAN and HISSCLU, very few algorithms exist in the realm of semi-supervised density-based clustering. Ruiz, Spiliopoulou and Ruiz (2007), Ruiz, Spiliopoulou and Menasalvas (2010) proposed C-DBSCAN, which is a modified version of DBSCAN designed to cope with instance-level constraints. However, C-DBSCAN has the same limitation as DBSCAN in that it uses a single, critical global density threshold determined by two user-defined parameters. In addition, the algorithm enforces constraints in a hard sense, i.e., clusters under cannot-link constraints are not allowed to be formed and different clusters under must-link constraints are forced to be merged. Hence, while satisfying the user-provided constraints, the algorithm violates the implicit assumptions behind the clustering model adopted, namely, the definitions of density connectivity and density-based clusters.

An algorithm of particular interest that does not suffer from any of the above limitations is HDBSCAN* (CAMPELLO; MOULAVI; SANDER, 2013; CAMPELLO *et al.*, 2015), described in Chapter 2. Originally, HDBSCAN* was proposed as a method for unsupervised or (constraint-guided) semi-supervised density-based clustering. In this thesis, we extend HDBSCAN* in two different ways: first, it is extended to also perform semi-supervised classification via label propagation; second, it is extended to perform semi-supervised clustering directly from labels, rather than instance-level pairwise constraints.

In the following section we discuss in more detail fundamental concepts and ideas underpinning the algorithms SSDBSCAN and HISSCLU, establishing the links between these algorithms and the ones described in Chapter 2, which will be subsequently required to understand our proposed unified approach for density-based semi-supervised clustering and classification.

3.1 SSDBSCAN

3.1.1 Conceptual Approach

SSDBSCAN (LELIS; SANDER, 2009) is a semi-supervised algorithm that performs semi-supervised clustering of an unlabeled dataset $\mathbf{X}_U \subset \mathbf{X}$ from a small fraction of labeled

data $\mathbf{X}_L \subset \mathbf{X}$ using a label expansion engine that is very similar to OPTICS. Unlike OPTICS, however, SSDBSCAN circumvents the unnecessary complications related to border objects and the asymmetric nature of the original reachability distance in Definition 6 by using the symmetric version, the *mutual reachability distance* (Definition 8) (LELIS; SANDER, 2009; CAMPELLO; MOULAVI; SANDER, 2013; CAMPELLO *et al.*, 2015)¹.

The interpretation of this definition plays a fundamental role not only in SSDBSCAN but also more broadly here in our work: the mutual reachability distance is the *smallest value of the radius ϵ* for which the corresponding pair of objects are still core objects and are ϵ -reachable from each other (Definition 2). As for a given m_{pts} , ϵ establishes a density threshold that is inversely proportional to this radius, it follows that the mutual reachability distance is then *inversely proportional to the largest density threshold* for which the corresponding pair of objects is directly density-connected according to Definition 3.

Conceptually, SSDBSCAN attempts to solve the following problem: for each unlabeled object, $\mathbf{x}_i \in \mathbf{X}_U$, the goal is to assign $\mathbf{x}_i$ the same label, $\text{class}(\mathbf{x}_j)$, as the object $\mathbf{x}_j \in \mathbf{X}_L$ that is the “closest” to $\mathbf{x}_i$ from a density-connectivity perspective, if such a labeling is possible, without violating the *label consistency* assumption (Assumption 5). This assumption requires that objects with different labels have to reside in disjunct density-based clusters following Definition 4 (w.r.t. the same m_{pts} , but possibly different ϵ values).

Let us provisionally put the label consistency assumption aside and first focus on the primary goal, namely, what “*closest to $\mathbf{x}_i$ from a density-connectivity perspective*” means. In density-based clustering, this refers to the object that can reach out (or be reached from) $\mathbf{x}_i$ through a path along which the lowest density connection is as high as possible (in other words, the *weakest* point in the connection is as strong as possible). Notice that, in light of Definition 8, this is equivalent to the path along which the largest *mutual reachability distance* is as small as possible. From a graph perspective, if objects are vertices and any two vertices are connected by an undirected edge weighted by their mutual reachability distance, the goal is to find the path between the unlabeled vertex $\mathbf{x}_i$ in question and a labeled vertex $\mathbf{x}_j$ along which the largest edge is minimal. Given $\mathbf{x}_i$ and any labeled candidate $\mathbf{x}_j$, this corresponds to the classic *minmax* problem in graph theory, whose solution can be proven to be the path between $\mathbf{x}_i$ and $\mathbf{x}_j$ along the minimum spanning tree (MST) of the graph. Hence, a preliminary approach to label the set of unlabeled objects $\mathbf{X}_U \subset \mathbf{X}$ could be the following:

Definition 11 (Label Propagation). Compute the MST of the dataset $\mathbf{X}$ in the transformed space of mutual reachability distances, MST_r , find the largest MST_r edge connecting each object $\mathbf{x}_i \in \mathbf{X}_U$ to every object $\mathbf{x}_j \in \mathbf{X}_L$, and make $\text{class}(\mathbf{x}_i) = \text{class}(\mathbf{x}_j)$, where $\mathbf{x}_j$ is the labeled object for which such a maximum edge is minimal.

¹ Originally, the *mutual reachability distance* was proposed in Lelis and Sander (2009), and used later in the definitions framework HDBSCAN*

Figure 10 – SSDBSCAN label assignment: in case (I), object $\mathbf{x}_i$ is assigned to $\mathbf{x}_{j2}$ (red); in case (II), $\mathbf{x}_i$ is left unlabeled as noise.

Source: Elaborated by the author.

From the density-based *clustering* perspective, however, there is a problem with this approach. Let us consider two labeled objects, $\mathbf{X}_L = \{\mathbf{x}_{j1}, \mathbf{x}_{j2}\}$, such that $\text{class}(\mathbf{x}_{j1}) \neq \text{class}(\mathbf{x}_{j2})$ and $\mathbf{x}_{j2}$ is on the MST_r path between $\mathbf{x}_{j1}$ and an unlabeled object $\mathbf{x}_i \in \mathbf{X}_U$. Now consider two possible scenarios:

- In the first scenario, the largest MST_r edge on the path between $\mathbf{x}_{j1}$ and $\mathbf{x}_i$, say 20, is located on the sub-path between $\mathbf{x}_{j1}$ and the intermediate object, $\mathbf{x}_{j2}$. The largest edge on the path between $\mathbf{x}_{j2}$ and $\mathbf{x}_i$ is then smaller than 20, say 10 (Figure 10-I). In this case, it is safe to make $\text{class}(\mathbf{x}_i) = \text{class}(\mathbf{x}_{j2})$ without violating the label consistency assumption because any threshold $\varepsilon \in [10, 20)$ can make $\mathbf{x}_i$ density-connected to $\mathbf{x}_{j2}$ (and, therefore, part of the same density-based cluster according to Definition 4) but not to $\mathbf{x}_{j1}$.
- In the second scenario, the largest MST_r edge on the path between $\mathbf{x}_{j1}$ and $\mathbf{x}_i$, say 20, is located on the sub-path between $\mathbf{x}_{j2}$ and $\mathbf{x}_i$, whereas the largest edge on the path between $\mathbf{x}_{j1}$ and $\mathbf{x}_{j2}$ is, say, 10 again (Figure 10-II). In this case, the largest edge on the path between $\mathbf{x}_i$ and the two labeled objects in question is the same, yet those two labeled objects have different labels. There is no threshold that can keep $\mathbf{x}_i$ density-connected to either $\mathbf{x}_{j1}$ or $\mathbf{x}_{j2}$ but not to both. The only way to split $\mathbf{x}_{j1}$ apart from $\mathbf{x}_{j2}$ is by a threshold $\varepsilon < 10$, but this also splits both $\mathbf{x}_{j1}$ and $\mathbf{x}_{j2}$ from $\mathbf{x}_i$. SSDBSCAN handles this label inconsistency scenario by not labeling $\mathbf{x}_i$ at all, leaving it unclustered as *noise*.

3.1.2 Algorithmic Approach

Algorithmically, SSDBSCAN does not pre-compute the MST_r . Instead, SSDBSCAN runs an OPTICS search starting from each labeled object and assigning the corresponding label temporarily to the unlabeled objects as they are found and processed by the OPTICS ordering traversal. However, this procedure, called label expansion, uses the mutual reachability distance in Definition 8, rather than the original, asymmetric version of OPTICS in Definition 6. The use of a symmetric distance makes OPTICS algorithmically identical to Prim's algorithm to compute MSTs. From this perspective, SSDBSCAN dynamically builds an MST in the space of mutual reachability distances, starting from each object $\mathbf{x}_{j1} \in \mathbf{X}_L$ and provisionally assigning its

label $\text{class}(\mathbf{x}_{j1})$ to the traversed unlabeled objects until an object $\mathbf{x}_{j2} \in \mathbf{X}_L$ with a different label, $\text{class}(\mathbf{x}_{j2}) \neq \text{class}(\mathbf{x}_{j1})$, is found. When such an object is found, SSDBSCAN backtracks and only confirms the labels assigned to objects before the largest edge on the path from $\mathbf{x}_{j1}$ to $\mathbf{x}_{j2}$, as these objects are “closer” to $\mathbf{x}_{j1}$ than to $\mathbf{x}_{j2}$ from a density-connectivity perspective, and they can be separated from those objects beyond such a largest edge (including $\mathbf{x}_{j2}$) by any threshold ϵ smaller than this edge’s weight.

SSDBSCAN stops when the label expansion procedure has been run from each $\mathbf{x}_j \in \mathbf{X}_L$. Objects not labeled by any of the OPTICS initializations are left unclustered as noise. These objects are those for which no density threshold exists that can make them part of any cluster without incurring label inconsistency.

3.1.3 Shortcomings

SSDBSCAN has a number of shortcomings that will be addressed later in this work:

1. **Order-dependency:** If the MST_r is not unique, i.e., when there are different yet equally optimal (minimum) spanning trees in the transformed space of mutual reachability distances, some objects may end up with different labels depending on the order of the various OPTICS traversals. The reason is that, for different traversals, the implicit minimum spanning trees that are partially and dynamically built from different starting vertices (labeled objects) may be different, and these differences can be shown to possibly cause the algorithm to be order-dependent. Order-dependency may also occur when there is more than one largest edge (i.e., a tie) on the sub-path of the MST_r between two labeled objects that have different labels;
2. **Re-computations:** rather than pre-computing the MST_r , SSDBSCAN implicitly and partially builds it from different starting vertices. Clearly, many portions of the MST_r are likely to be recomputed multiple times, which is unnecessarily inefficient from a computational point of view;
3. **Noise and missing clusters:** from a *clustering* perspective, the fact that some objects are left unclustered as noise is expected, especially from a density-based perspective. In SSDBSCAN, however, entire clusters may be left unclustered as noise, typically when they do not contain any labeled object and are well-separated from other clusters, closer to each other, containing objects with different labels, such that the missed clusters cannot be reached at a density level without incurring violations of the label consistency assumption (Assumption 5). This is why SSDBSCAN assumes that there is “at least one (possibly more) labeled object from each class” (Assumption 4), which is, however, a *classification* rather than a clustering assumption. From a classification perspective, even when no cluster is missed, leaving a fraction of objects unlabeled as noise (typically global or local outliers)

may be undesired. This particular issue is not present in a related algorithm, HISSCLU, which we discuss in the next section.

3.2 HISSCLU

HISSCLU (BÖHM; PLANT, 2008) can be seen as a semi-supervised version of OPTICS that produces, instead of the original, purely unsupervised reachability plot, a colored version of the plot where every unlabeled object of the dataset is assigned a class label (i.e., a color) from the labeled set, $\mathbf{X}_L \subset \mathbf{X}$. The colored plot provides a visual contrast between the clustering structure, revealed as valleys and peaks following an OPTICS ordering ($\prec$), and the transductive classification, mapped as colors in the plot. The main algorithm consists of two stages, a **preprocessing stage** and a **label expansion stage**, as described next.

3.2.1 Preprocessing stage (label-based distance weighting)

Before any label expansion takes place, HISSCLU pre-computes a weight for each pairwise distance $d(\mathbf{x}_i, \mathbf{x}_j)$ in the dataset. The resulting, weighted distances are used in lieu of the original (e.g., Euclidean) distances in the subsequent steps of the algorithm, namely, reachability distance computations and OPTICS-based label propagation. Such a preprocessing stage is designed to widen the gap between nearby classes by stretching distances between objects around class boundaries, thus allowing the algorithm to expand different class labels even in situations where no natural boundaries of low density between different classes exist. The intuition of the algorithm is described as following: a pair of objects $(\mathbf{x}_p, \mathbf{x}_q) \in \mathbf{X}_L \times \mathbf{X}_L$ for which $\text{class}(\mathbf{x}_p) \neq \text{class}(\mathbf{x}_q)$ establishes a separating hyperplane that perpendicularly crosses the midpoint on the line segment between these two objects (as points in an Euclidean space). A pair of objects $(\mathbf{x}_i, \mathbf{x}_j)$ on different sides of this hyperplane in the vicinity of $\mathbf{x}_p$ and $\mathbf{x}_q$ will have their distance stretched by a multiplicative weight ≥ 1 . The closer to the hyperplane, the greater the weight (stretching). The maximum weight is a user-defined parameter, $\rho \geq 1$ ($\rho = 1$ means no weighting). The weighting decays towards the minimum value 1 with an increasing distance of the objects from the separating hyperplane. Objects “behind” $\mathbf{x}_p$ and $\mathbf{x}_q$ in relation to the hyperplane are not affected (unitary weight). The shape and rate of decay is controlled by a second parameter, $\xi > 0$ ($\xi = 1$ gives a parabolic decrease, $\xi > 1$ gives a faster, bell-shaped decrease corresponding to a sharper “influence region” around the separating hyperplane, whereas $0 < \xi < 1$ gives a more square-shaped decrease, i.e., a wider influence region around the separating hyperplane). This mechanism allows HISSCLU to expand different class labels even within clusters that are formed by more than one class, i.e., clusters of objects that are density connected but not pure in their labels.

Figure 11 – Label assignment in HISSCLU: object $\mathbf{x}_2$ is assigned to $\mathbf{x}_1$ (green), whereas $\mathbf{x}_3$ and $\mathbf{x}_4$ are assigned to $\mathbf{x}_5$ (red).

Source: Elaborated by the author.

3.2.2 Label expansion stage

Conceptually, the transductive classification performed by HISSCLU is essentially the label propagation procedure described in Definition 11, which also serves as a basis for SSDBSCAN as previously discussed in Section 3.1. However, unlike SSDBSCAN, HISSCLU does not make the label consistency assumption (Assumption 5), thus being able to assign a label to every unlabeled object in $\mathbf{X}_U \subset \mathbf{X}$. In other words, no object is left unlabeled as noise.

But if noise is not an option, how does HISSCLU handle possible label inconsistencies, which can be caused by ties w.r.t. the maximum MST_r edge (referred to in the definition)? For instance, let us consider the illustrative example in Figure 11, where we have 5 data objects and the corresponding MST_r . The label propagation procedure described in Definition 11 would arguably assign $\text{class}(\mathbf{x}_2) = \text{class}(\mathbf{x}_1)$ (green) and $\text{class}(\mathbf{x}_4) = \text{class}(\mathbf{x}_5)$ (red). The middle point ($\mathbf{x}_3$), however, is undefined, as there is a tie in the largest edge (10) from this point to objects with different labels. In SSDBSCAN, this point should be left unlabeled (noise), because there is no density threshold that can keep it density-connected to either $\mathbf{x}_1$ or $\mathbf{x}_2$ but not to both.² Differently, HISSCLU uses instead the 2nd largest edge to resolve the tie (then the 3rd if there is another tie, and so on). In our example, the 2nd largest edge is 9 (green) vs. 5 (red), thus $\text{class}(\mathbf{x}_3) = \text{class}(\mathbf{x}_5)$ (red).

Algorithmically, HISSCLU operates similarly to SSDBSCAN in the sense that it also runs an OPTICS search starting from each labeled object in $\mathbf{X}_L$ and assigning labels temporarily to unlabeled objects as they are found by the OPTICS ordering traversal. Unlike SSDBSCAN, however, the traversal does not stop and backtracks once an object with a different label is found. More importantly, the OPTICS traversal in HISSCLU occurs *simultaneously* from all labeled objects. To that end, the OPTICS priority queue is initialized with every unlabeled object having its smallest reachability distance from a labeled object, say $\mathbf{x}_j \in \mathbf{X}_L$, as its priority key, and $\text{class}(\mathbf{x}_j)$ as its temporary label. This corresponds to initializing OPTICS from all objects in $\mathbf{X}_L$ at the same time, rather than from a single object. Each object $\mathbf{x}_o$ removed from the queue is processed and its temporary label becomes permanent. The reachability distance from $\mathbf{x}_o$ to

² Algorithmically, however, $\mathbf{x}_3$ may in practice be labeled depending on how the tie is resolved in the backtracking step of the label expansion procedure, and the final label may in this case be order-dependent.

every object $\mathbf{x}_i$ still unprocessed inside the queue is computed, and whenever $\mathbf{x}_i$ is reached by a processed object $\mathbf{x}_o$ with reachability distance smaller than $\mathbf{x}_i$'s current priority key, $\mathbf{x}_i$'s key is updated, its temporary label is set to $\text{class}(\mathbf{x}_o)$, and the priority queue is rearranged accordingly. This procedure has been shown (BÖHM; PLANT, 2008) to ensure that the final labels respect the desired “min-max” reachability notion, resolving ties in the maximum edge as described above. However, HISSCLU is still subject to issues previously discussed in the context of SSDBSCAN (Section 3.1.3), which relate to the fact that (possibly different) minimum spanning trees are built partially and dynamically, rather than pre-computed.

3.2.3 Flat Clustering Extraction (*k*-clustering)

Like OPTICS, the reachability plot that results from HISSCLU encodes only visually and *implicitly* a density-based clustering hierarchy. The colors in the plot, in turn, represent a transductive classification of the data from the collection of pre-labeled objects, rather than a clustering result. For scenarios where an explicit clustering solution is desired, Böhm and Plant (2008) offer an optional, post-processing flat clustering extraction stage of HISSCLU, called *k*-clustering, which essentially applies an arbitrary global density threshold to perform a conventional horizontal cut through the reachability plot, analogous to extracting DBSCAN solutions from OPTICS.

Algorithmically, to perform the cluster extraction in HISSCLU, the authors Böhm and Plant (2008) consider d_{max} as the maximum reachability distance in an ordered file, and consider that clusters are separated by a distance of at least $d_{sep} = k \cdot d_{max}$, with $k \in [0, 1]$. Besides, they also require that each cluster has at least m_{pts} objects. The extraction occurs by searching a sequence of n objects ($n \geq m_{pts}$) in the reachability plot, in which the first object of the sequence has the reachability distance higher than d_{sep} and the remaining objects have reachability distance less than d_{sep} .

3.3 Final remarks

In this Chapter we present the main characteristics of semi-supervised learning. We describe the links and the differences between semi-supervised classification and semi-supervised clustering. At least, we identify, under assumptions of semi-supervised classification algorithms, that the algorithms SSDBSCAN and HISSCLU originally developed for semi-supervised clustering are, in fact, algorithms more suitable for semi-supervised classification task. We identify shortcomings for these algorithms such as order-dependency and multiple construction of the Minimum Spanning Tree during their execution. We will use these as base for the development of a novel framework for semi-supervised classification and for comparison in the scenario of the label-based semi-supervised clustering algorithm. In the next chapter, we present an overview of compound mining and semi-supervised algorithms applied to the area.

COMPUTER-AIDED DRUG DESIGN

Research in the Medicinal Chemistry area (MC) aims to investigate the main mechanisms related to the development of a disease to propose new chemical candidates for medicines. To do this, researchers are responsible for validating a biological target, discovering or designing new compounds, conducting biological tests, among other activities (YOUNG, 2009). Currently, there is an intensive application of machine learning and data mining techniques, since they reduce financial and time costs in the drug discovery process (YOUNG, 2009; GERTRUDES *et al.*, 2012; MALTAROLLO, 2013).

Figure 12 – Drug discovery and development process.

Source: Elaborated by the author.

The process of discovering and developing new drugs follows, in general, a common set of steps, as shown in Figure 12. Once the disease has been identified (step 1 of Figure 12), it is necessary to know its biological target (target validation) which, when interacting with a particular drug, changes the metabolism of an individual, alleviating or healing the disease (YOUNG, 2009).

A biological target may be a protein, receptor, or enzyme which, when interacting with a particular compound, modifies the individual's metabolism, alleviating disorders in the organism. In the drug development, it is necessary to understand in detail the interactions between the substance and the biological target related to specific diseases, so that the stimulus is carried out

in the most efficient way possible (MALTAROLLO, 2009).

Most existing drugs have a biological effect dependent on a specific interaction between a molecule and a biological target. This interaction can be described by the lock key theory (FISCHER, 1894), as shown in Figure 13. Figure 13-I presents the interaction between a substrate (“key”) and the biological target (“lock”), performing an efficient interaction and triggering the biological response. Figure 13-II presents an interaction between an incorrect substrate and an enzyme that blocks the biological response.

Figure 13 – Possible interactions between a bioactive substance and its biological target.

Source: Elaborated by the author.

During the drug discovery process, MC researchers analyze whether a particular biological target needs to have its biological response activated or blocked (step 2 of Figure 12). This decision proceeds after a detailed study of the biological target and its action in the organism. To quantify the level of interaction between a molecule and its biological target, researchers in MC use indices of biological activity (GERTRUDES, 2013).

After identifying the biological target, the investigation of molecules for drug candidates is performed (step 3 in Figure 12), which can be performed either by analyzing the structure-activity relationships (SAR) or through virtual screening of molecules (TUCCINARDI, 2009; MALTAROLLO, 2013). In the first method, a database of molecules is used that already has information about their interaction with the biological target (active and inactive), whereby multivariate data analysis methods are used to establish the SAR. In virtual screening, a computer simulation is performed with the orientation of a molecule with its biological target to evaluate whether, when attached, they form a stable connection.

This document focuses on SAR analysis because this technique selects new molecules spending less money, both computationally and financially. In this analysis, data sets with observations (molecules) and attributes (chemical descriptors) are evaluated, containing information about the properties of bioactive substances. They are analyzed, in general, with the application of classification or regression methods (KITCHEN; STAHURA; BAJORATH, 2004; GERTRUDES *et al.*, 2012). Once the SAR relationships have been established, MC researchers perform the modeling of new molecules and use the rules obtained with the classification/regression tech-

niques to predict the interaction of a new candidate compound with the biological target under study.

The SAR analysis process can be divided into two steps: molecular modeling and chemometrics. In the first case, molecules with known biological activity are selected through queries in the literature or through database searches (e.g, ZINC database ¹). After this choice, transformations of these molecules in a dataset with observations and attributes are performed, which will represent chemical properties of each molecule. There are tools such as SPARTAN² and DRAGON³ that can calculate these attributes descriptors. With the matrix of observations and attributes, multivariate analysis (also known as chemometric analysis) are performed, and involve the selection of the best chemical descriptors by means of dimensionality reduction techniques, outlier detection, application of machine learning techniques, and the validation of the obtained models.

Once the interaction of a new molecule has been screened or targeted to its target, bioassays are conducted in order to validate its biological potential. After its biological potential has been confirmed, the molecule follows on for the optimization step and pre-clinical tests (steps 4 and 5 of Figure 12)⁴. Finally, after the pre-clinical tests, the candidate drug follows to the last two steps of the drug development process, which correspond to the clinical trials and approval by a regulatory agency (steps 6 and 7 of Figure 12).

4.1 SAR in Medicinal Chemistry

One of the main difficulties of applying classification/regression techniques for SAR is the low number of labeled molecules. According to the authors Ning and Karypis (2011), Andonie (2010), and Varnek and Baskin (2012), most chemicals do not have pre-defined labels and samples of labeled compounds are not large enough to generate a good classification/regression model. Besides, according to Digles and Ecker (2011), some data sets do not have a clear separation between active and inactive molecules. For the above reasons, the application of techniques that can identify relationships between similar molecular and biological activity should be recommended, even in the scenarios described above.

In a study conducted by Gertrudes (2013), the application of three clustering algorithms was carried out to scout for relationships between chemical structure and biological activity. The clustering algorithms were applied to four data sets containing information about bioactive substances. In this case, the label information⁵ of these sets was used for external (controlled)

¹ Database available at <http://zinc.docking.org>

² Available at <https://goo.gl/xHCCvC>

³ Available at <https://goo.gl/SyHYSX>

⁴ The information about the predicted molecule can also be used in screening and prediction of new molecules

⁵ Represented by a molecule as active or inactive, according to a threshold defined by domain expert.

validation of groups obtained by the aforementioned unsupervised clustering algorithms. In the experiments, the results encourage the use of these algorithms in the clustering molecules task.

One of the issue that can be improved in the work of [Gertrudes \(2013\)](#) is that, in practice, when a certain (typically small) fraction of labeled objects is available, this information should be incorporated into the learning process, increasing its power. Semi-supervised learning algorithms have a good chance to improve SAR studies since they are specially designed for these scenarios.

It is noteworthy that recently, there has been a significant increase in the number of robust computational methods used in data analysis in MC, such as artificial neural networks and SVM ([GERTRUDES *et al.*, 2012](#)). In semi-supervised scenarios, [Kondratovich, Baskin and Varnek \(2013\)](#) show that Transductive Support Vector Machines (TSVM) can be particularly useful to build models on small and unbalanced datasets which often represent a difficult problem in quantitative SAR (QSAR) studies.

A second issue that can be improved with respect to the study of [Gertrudes \(2013\)](#) concerns feature selection. The selection of features that mathematically represent the compound set and the relationships with the biological activity are not a trivial task. The methods used to generate the molecular descriptors are able to provide a large number of variables (some of them may have up to thousands of descriptors). Furthermore, there is an ideal condition in SAR studies: in general, each descriptor can explain five chemical compounds ([MALTAROLLO, 2009](#); [ARAÚJO *et al.*, 2015](#)). This proportion, called the parsimony principle, was proposed to facilitate the physicochemical interpretation of the SAR model and also to avoid the overfitting in SAR modeling ([MALTAROLLO, 2009](#); [ARAÚJO *et al.*, 2015](#)).

To perform the dimensionality reduction in QSAR studies, [Hasanloei *et al.* \(2018\)](#) extended the Fisher score to define the new distance metrics based on the continuous activity values of compounds with known activities. Then, a semi-supervised feature selection method was proposed based on the combination of Fisher and Laplacian criteria which exploits both compounds with known and unknown activities to select the relevant descriptors. The results demonstrated that the QSAR models built on the selected descriptors by the proposed semi-supervised method perform better than traditional methods for feature selection.

4.2 Final remarks

In this chapter, we defined some concepts used in the course of this document, such as the drug development process, biological target, biological activity, in addition to the analysis between chemical structure and biological activity (SAR). With respect to the discovery of new drug candidates, it can be observed that this process is very costly for researchers in MC and their time of analysis can be reduced with the aid of computational methods.

UNIFIED FRAMEWORK FOR DENSITY-BASED CLASSIFICATION

In the previous chapters, we elaborated how previous clustering and semi-supervised clustering algorithms in the density-based clustering paradigm can all conceptually be viewed as processing minimum spanning trees in a space of reachability distances, i.e., processing MSTs of a conceptual, complete graph where the nodes are the objects, and the edge weights are the reachability distances between objects. These reachability distances can be based on unmodified or modified (e.g., weighted, stretched) distances between objects.

In this chapter, we present a new framework for semi-supervised classification by extending the HDBSCAN* clustering framework with additional, optional steps, derived from “decoupled” building blocks of the algorithms discussed in [Chapter 3](#) so that these building blocks can be re-combined and applied in different ways. This will allow us to study the performance gain of each building block and specify different instances for semi-supervised classification — one can be considered a close approximation of HISSCLU, and another one is a looser, but faster approximation, others are novel variants that have not been investigated before.

5.1 The Components of the Framework (Building Blocks)

The building blocks for semi-supervised classification in our framework are the following:

1. **The adopted definition of core and reachability distances:** In our framework, we will study both the standard definition of core-distance with the parameter m_{pts} , which has been adopted by all the algorithms discussed in [Chapter 2](#) (Definition 5), as well as the parameter-free all-points core distance (Definition 10). Given a notion of core distance, we will only use the symmetric notion of *mutual* reachability distance (Definition 8) as used

by SSDBSCAN, DBSCAN*, and HDBSCAN*, even though the “older” algorithms HISSCLU, DBSCAN, and OPTICS are based on the asymmetric notion in Definition 6. The reason is that the mutual reachability distance has a statistically more sound interpretation, is simpler and, in practice, the difference in results tends not to be very noticeable.

2. **MST computation in the space of mutual reachability distances:** Recall from Chapter 3 that SSDBSCAN and HISSCLU compute multiple MSTs “on-the-fly”, starting from labeled objects. Such an approach is inefficient and, from a conceptual point of view, not necessary. It is obviously not necessary when the MST of the conceptual, complete graph in the transformed space of mutual reachability distances is unique; then the same MST will just be re-computed multiple times. But even when the MST is not unique, the different MSTs are in a sense equivalent from the perspective of representing the inherent cluster structure of a data set: none of them should lead to fundamentally different conclusions about the density distribution of the dataset. Therefore, in our framework we explicitly decouple the MST construction from the label expansion (as we have already done conceptually in the discussion of the algorithms SSDBSCAN and HISSCLU in the previous chapter), and we will use HDBSCAN*’s efficient algorithm to compute the “extended” MST_r as described in [subsection 2.3.2](#).
3. **Label expansion:** Given a computed graph MST_r , in our framework for semi-supervised classification we implement a label expansion method on top of the MST_r , similar to HISSCLU, but using the single MST_r computed by HDBSCAN* to propagate the labels based on the path with the smallest largest edge to a labeled object, resolving ties, possibly consecutively, by considering the smaller of the next largest edge on the paths (see [subsection 3.2.2](#)).

This can be implemented by starting with a “connected component” C_i for each pre-labeled object $\mathbf{x}_i \in \mathbf{X}_L$ that initially contains only $\mathbf{x}_i$. These connected components C_i are then iteratively extended by traversing the MST_r in the following way. The outgoing edges from each C_i in the MST_r are maintained in a sorted order, from smallest to largest edge weight. Then, always the smallest edge weight of all such edges (connecting to any of the C_i) is determined, and all of those edges with this edge weight are selected simultaneously. First, the selected edges that establish a connection between the current connected components will be considered: components that have the same label and are connected with selected edges will be merged into a single connected component that replaces them, whereas selected edges between connected components with different labels are ignored. Then, all objects $\mathbf{x}_o \in \mathbf{X}_U$ that connect to the selected edges and that have not been labeled yet will be considered in the following cases:

- a) only a single edge, i.e., a single current connected component C_i connects to an object $\mathbf{x}_o$; in this case, $\mathbf{x}_o$ and the corresponding edge are added to C_i and $\text{class}(\mathbf{x}_o)$ is set to $\text{class}(C_i)$;

- b) more than one edge, i.e., components $C_{i_1}, \dots, C_{i_m}$ connect to object $\mathbf{x}_o$; in this case, two sub-cases are possible:
- i. all the components connected to $\mathbf{x}_o$ have the same label. In this sub-case all these components (including $\mathbf{x}_o$ and the corresponding edges) are merged into a single component;
 - ii. the components connecting to $\mathbf{x}_o$ have different labels. In this sub-case, the algorithm determines the component C_{i_j} that has the smallest largest edge (or, in case of ties, the smallest of the next largest edge, *etc.*) on a path to $\mathbf{x}_o$ from a pre-labeled object inside that component, $\mathbf{x}_o$ and its edge are added to C_{i_j} , $\text{class}(\mathbf{x}_o)$ is set to $\text{class}(C_{i_j})$, and all other components connecting to $\mathbf{x}_o$ that have the same class label as $\text{class}(C_{i_j})$ are merged with C_{i_j} (other components with different labels are not merged in this step, even if some of them share a label).

In both cases (a) and (b), the still unprocessed edges of the MST_r incident to $\mathbf{x}_o$ (if any) will be added to the sorted set of outgoing edges from the current components, and the sorted set will be rearranged accordingly. If none of the selected edges in an iteration fit into one of the above scenarios, simply the next smallest edge weight from the sorted set is selected, and this process is repeated until all edges in the MST_r have been processed (each edge will be processed once).

If a result equivalent to SSDBSCAN's label assignments would be desired, the result of the label expansion could be post-processed to remove propagated labels from objects that lead to label inconsistencies, declaring them as Noise (see Section 3.1). However, since a label assignment for all $\mathbf{x} \in \mathbf{X}_U$ is required in the context of semi-supervised *classification*, such a noise identification procedure will not be further discussed here.

The information of the smallest largest edge connecting a given unlabeled object $\mathbf{x}_o$ to one of the pre-labeled objects inside a given component C_i can be efficiently stored in the objects themselves and expanded alongside with the labels. Let us suppose that an object $\mathbf{x}_o \in \mathbf{X}_U$ has just been added to component C_i via an edge $(\mathbf{x}_o, \mathbf{x}_i)$ connecting $\mathbf{x}_o$ to $\mathbf{x}_i \in C_i$. Let $w_{o,i}$ be this edge's weight, and let w_i be the smallest largest edge weight connecting $\mathbf{x}_i$ to a pre-labeled object inside C_i ($w_i = 0$ if $\mathbf{x}_i$ itself is pre-labeled, i.e., if $\mathbf{x}_i \in \mathbf{X}_L$). Then, the smallest largest edge weight on the path from $\mathbf{x}_o$ to a pre-labeled object inside C_i can be stored in $\mathbf{x}_o$ as $w_o = \max\{w_i, w_{o,i}\}$. These values will be stored in objects in the frontier of the components as these components grow. By doing this, the components C_i do not need to be materialized, they will be implicitly encoded in the expanded labels.

Notice that, if an unlabeled object $\mathbf{x}_o$ is simultaneously connected to more than one component (cases (b)-i or (b)-ii above), say C_i and C_j , via selected edges $(\mathbf{x}_o, \mathbf{x}_i)$ and $(\mathbf{x}_o, \mathbf{x}_j)$ with weights $w_{o,i} = w_{o,j}$, then the value w_o to be stored in $\mathbf{x}_o$ will be the smaller of $\max\{w_i, w_{o,i}\}$ and $\max\{w_j, w_{o,j}\}$ (i.e., the smallest largest edge weight connecting $\mathbf{x}_o$

to one of the pre-labeled objects either inside C_i or inside C_j). The smaller of these values will also decide the label of $\mathbf{x}_o$ in case $\text{class}(C_i) \neq \text{class}(C_j)$ (i.e., case (b)-ii). In order to solve ties ($\max\{w_i, w_{o,i}\} = \max\{w_j, w_{o,j}\}$), instead of keeping only the smallest largest edge weight stored in each object, we keep a sorted list of the c largest weights on the respective paths, where c is a constant. This way, we do not need to traverse the MST_r to decide which component has the smallest next largest edge in case of ties. Only if c values are tied a traversal would be required. But this is unlikely in practice, and it would mean that the label is highly undecided, so breaking the tie randomly is nearly as accurate yet much faster. Since c is a constant (we use $c = 5$ in our experiments), it does not affect the algorithm's runtime or memory complexity.

Example:

Figure 14 illustrates our label expansion with an example. Figure 14a displays 15 objects and their minimum spanning tree in the mutual reachability distance space (MST_r). To perform the label expansion, we initialize connected components with the pre-labeled objects, $C_1 = \{\mathbf{x}_1\}$, $C_2 = \{\mathbf{x}_5\}$, $C_3 = \{\mathbf{x}_9\}$, and $C_4 = \{\mathbf{x}_{13}\}$.

- a) From all the components, the first edge to be analyzed is the one connecting $\mathbf{x}_5$ to $\mathbf{x}_6$ since it is the one with the lowest weight, 1.0, among all of the currently “outgoing” edges of the current components. Since $\mathbf{x}_6$ is not labeled yet, it will receive the label of C_2 (i.e., $\text{class}(\mathbf{x}_5)$), $\mathbf{x}_6$ is added to C_2 , and since there is no other edge incident to $\mathbf{x}_6$, no new outgoing edges are added.
- b) Then, the edges with the next largest edge weight, 1.5, connecting to C_1 and to C_4 , which are the edges connecting $\mathbf{x}_1$ to $\mathbf{x}_3$ and connecting $\mathbf{x}_{13}$ to $\mathbf{x}_{14}$, are selected. The two edges connect to a different, not-yet-labeled object and thus $\mathbf{x}_3$ is labeled with $\text{class}(\mathbf{x}_1)$ and added to C_1 , while $\mathbf{x}_{14}$ is labeled with $\text{class}(\mathbf{x}_{13})$ and added to C_4 .
- c) Two new unprocessed edges are now incident to C_1 , connecting $\mathbf{x}_2$ and $\mathbf{x}_4$ to $\mathbf{x}_3$, with edge weights of 4.0 and 1.0, respectively; three new unprocessed edges are now incident to C_4 , connecting $\mathbf{x}_{11}$, $\mathbf{x}_{12}$, and $\mathbf{x}_{15}$ to $\mathbf{x}_{14}$, with edge weights of 2.5, 4.0, and 2.0, respectively. The smallest edge weight of all the outgoing edges of the current connected components is now 1.0 on the edge connecting $\mathbf{x}_4$ to $\mathbf{x}_3$. Since $\mathbf{x}_4$ is unlabeled, it is labeled with $\text{class}(\mathbf{x}_1)$ and added to C_1 , and the edge connecting $\mathbf{x}_4$ to $\mathbf{x}_5$ is added to the outgoing edges of C_1 .
- d) The now smallest edge weight of outgoing edges is 2.0 on the edges connecting $\mathbf{x}_9$ to $\mathbf{x}_8$ and connecting $\mathbf{x}_{14}$ to $\mathbf{x}_{15}$. The two edges are incident to two different unlabeled objects. Hence $\mathbf{x}_8$ is added to C_3 (which currently had only $\mathbf{x}_9$ in it) and receives its label, and $\mathbf{x}_{15}$ is added to C_4 (currently with $\mathbf{x}_{13}$ and $\mathbf{x}_{14}$) and receives its label; no new outgoing edges are incident to C_4 but three new outgoing edges are incident to

Figure 14 – Example of density-based label expansion.

(a) MST_r , with 4 pre-labeled objects ($\mathbf{X}_L = \{x_1, x_5, x_9, x_{13}\}$) from 3 classes (red, blue, green)

(b) Scenario where there is tie edges (largest edge connecting to a component with different labels.)

(c) Result of the label expansion

Source: Elaborated by the author.

- C_3 : $(\mathbf{x}_8, \mathbf{x}_7)$, $(\mathbf{x}_8, \mathbf{x}_5)$, and $(\mathbf{x}_8, \mathbf{x}_{10})$, with corresponding edge weights 1.0, 3.0, and 4.0.
- e) Next, the smallest edge weight of outgoing edges is now 1.0 on the edge $(\mathbf{x}_8, \mathbf{x}_7)$; $\mathbf{x}_7$ is added to C_3 and receives its label, no new outgoing edges are incident to $\mathbf{x}_7$. Similarly, $\mathbf{x}_{11}$ is added to C_4 since its edge weight of 2.5 is the next smallest.
 - f) After adding $\mathbf{x}_{11}$ to C_4 , C_4 has a new outgoing edge, connecting $\mathbf{x}_{11}$ to $\mathbf{x}_{10}$, with edge weight 4.0. Next, the smallest edge weight of outgoing edges is now 3.0 on the edge between $\mathbf{x}_5$ and $\mathbf{x}_8$, outgoing from C_2 as well as C_3 . This edge connects two components with the same label. Hence C_2 and C_3 are merged into and replaced by C_{2_3} .
 - g) In the next step the smallest edge weight of outgoing edges is 4.0 on the edges $(\mathbf{x}_3, \mathbf{x}_2)$, outgoing from C_1 , $(\mathbf{x}_8, \mathbf{x}_{10})$, outgoing from C_{2_3} , $(\mathbf{x}_{11}, \mathbf{x}_{10})$ outgoing from C_4 , and $(\mathbf{x}_{14}, \mathbf{x}_{12})$, outgoing from C_4 . This scenario is presented in Figure 14b. Similar to previous cases in which an unlabeled object is connected to just one of the current components, $\mathbf{x}_2$ is labeled and added to C_1 , and $\mathbf{x}_{12}$ is labeled and added to C_4 .
 - h) The unlabeled object $\mathbf{x}_{10}$ is connected to two components with different class labels. The smallest largest edge weight on paths to pre-labeled objects in both C_{2_3} and C_4 is 4.0, so the next (2nd) largest edge on all paths to pre-labeled objects is considered, which are 2.0 on the path from $\mathbf{x}_9$ in C_{2_3} to $\mathbf{x}_{10}$, 3.0 on the path from $\mathbf{x}_5$ in C_{2_3} to $\mathbf{x}_{10}$, and 2.5 on the path from $\mathbf{x}_{13}$ in C_4 to $\mathbf{x}_{10}$. The smallest of these is 2.0 from $\mathbf{x}_9$, hence $\mathbf{x}_{10}$ is added to C_{2_3} and obtains its label.
 - i) In the last step, the edge $(\mathbf{x}_4, \mathbf{x}_5)$ with edge weight 5.0 is ignored since it connects two components with different class labels.

Figure 14c shows the final result.

4. **Preprocessing: Label-based distance weighting:** As an optional step, one may want to compute weighted distances based on the labeled subset of the data, as described in Section 3.2.1 for HISSCLU. We include such a step in our framework that applies label-based distance weighting on all the pairwise distances before computing core and reachability distances, as in HISSCLU. However, this step can be computationally time consuming. As an approximation, we also propose to apply distance weighting *after* the MST_r has been constructed, so that it only needs to be applied to the mutual reachability distances of the edges in the MST_r .

5.2 The Framework

The above building blocks can be combined in different ways to obtain different and novel semi-supervised classification methods as instances of our framework, through which we

Figure 15 – Unified framework of density-based algorithms for semi-supervised classification.

Source: Elaborated by the author.

can study the contribution of each building block in an overall approach to semi-supervised classification. We denote different instances using the notation $\text{HDBSCAN}^*(\text{core-distance-definition}, \text{label-based-distance-weighting-scheme})$, where *core-distance-definition* stands either for the standard core distance definition, abbreviated by “cd”, or the all-points core distance, abbreviated by “ap”; and *label-based-distance-weighting-scheme* stands for label-based distance weighting of all pairwise distances, abbreviated as “wPWD”, or label-based distance weighting of the MST_r edges, abbreviated as “wMST”; no weighting is denoted as “—”.

Figure 15 presents a schematic view of the instances of our framework for semi-supervised classification, which we will study in the experimental evaluation. Each branch in the diagram starts with a distance matrix as input and represents a different algorithm. In the leftmost branch, label-based distance weighting on the pairwise distances of the distance matrix is performed, standard core-distance is used, MST_r is computed, and label expansion is performed. This algorithm is denoted by $\text{HDBSCAN}^*(\text{cd}, \text{wPWD})$, and it is similar to HISSCLU,

but it uses the symmetric, mutual reachability distance rather than the old, asymmetric version, and relies only on a single pre-computed MST_r . The third branch that uses standard core distance and performs label-based distance weighting only on the MST_r edges can be considered an even looser but much faster approximation of HISSCLU. The other branches show other methods, using alternatively the all-points core distance, and perform or omit completely the two options for label-based distance weighting.

5.3 Complexity

The asymptotic complexity of instances of our framework is as follows. Given the dataset $\mathbf{X}$, computing the *core* or *all-points-core* distances and the construction of the MST_r has an overall time complexity of $O(n^2)$ (this part is the same as HDBSCAN*). The label expansion takes, in the worst case, $O(n \log n)$ time if the set of outgoing edges of the connected components is maintained in a heap, with edge weights as priority key.

- In the algorithms that apply the label-based weighting function to the entire distance matrix — HDBSCAN*(cd,wPWD) and HDBSCAN*(ap,wPWD) — the additional runtime is of the order $O(n^2 + |\mathbf{X}_L|n)$, where $|\mathbf{X}_L|$ is the number of pre-labeled objects: the first term is the number of distances that have to be weighted, whereas the second term corresponds to the pre-computation of the elements required to compute any weight in constant time (following the optimized approach of (BÖHM; PLANT, 2008)).
- For the algorithms that apply the weighting function in the MST_r instead — HDBSCAN*(cd,wMST) and HDBSCAN*(ap,wMST) — the additional runtime is $O(n + |\mathbf{X}_L|n) \rightarrow O(|\mathbf{X}_L|n)$ since the MST_r has only $n - 1$ edges to be weighted. Assuming $|\mathbf{X}_L| \ll n$ as usual in semi-supervised classification, the additional runtime of this approach is $O(n)$, in contrast to $O(n^2)$ of the original HISSCLU weighting. The former becomes even more attractive when it is necessary to compute the label expansion with different sets of labeled objects. In this case, it will be necessary only to adapt the edges of the MST_r , instead of repeating the process of computing the core distance (or the all-points core distance), and to compute the MST_r in every different label configuration.

In total, the overall runtime complexity of the algorithms in the framework for semi-supervised classification is hence $O(n^2)$. If pairwise distances are computed on demand, it requires $O(n)$ memory only.

5.4 Experimental setup

In this section we describe the experimental setup for the assessment of our proposed density-based methods for semi-supervised classification in terms of effectiveness as well as

Table 3 – List of real datasets collected to perform the semi-supervised classification experiments.

Dataset	#obj	#att	#cl	Distance
ACE ECFP4 (SUTHERLAND; O'BRIEN; WEAVER, 2004)	114	1,025	2	Tanimoto
ACE ECFP6 (SUTHERLAND; O'BRIEN; WEAVER, 2004)	114	1,025	2	Tanimoto
Analcatdata authorship (VANSCHOREN <i>et al.</i> , 2014)	841	70	4	Cosine
Armstrong-v1 (SOUTO <i>et al.</i> , 2008)	72	1,082	2	Cosine
Auto price (VANSCHOREN <i>et al.</i> , 2014)	159	16	2	Euclidean
Bank note–Authentication (VANSCHOREN <i>et al.</i> , 2014)	1,372	5	2	Euclidean
Cardiotocography (VANSCHOREN <i>et al.</i> , 2014)	2,126	36	10	Euclidean
Chowdary (SOUTO <i>et al.</i> , 2008)	104	183	2	Cosine
Chcase Geyser1 (VANSCHOREN <i>et al.</i> , 2014)	222	2	2	Euclidean
COX2 ECFP6 (SUTHERLAND; O'BRIEN; WEAVER, 2004)	322	1,025	2	Tanimoto
DHFR ECFP4 (SUTHERLAND; O'BRIEN; WEAVER, 2004)	397	1,025	2	Tanimoto
DHFR ECFP6 (SUTHERLAND; O'BRIEN; WEAVER, 2004)	397	1,025	2	Tanimoto
Diggle table (VANSCHOREN <i>et al.</i> , 2014)	310	8	9	Euclidean
Fontaine ECFP4 (FONTAINE <i>et al.</i> , 2005)	435	1,024	2	Tanimoto
Fontaine ECFP6 (FONTAINE <i>et al.</i> , 2005)	435	1,024	2	Tanimoto
Gordon (SOUTO <i>et al.</i> , 2008)	181	1,627	2	Cosine
Iris (LICHMAN, 2013)	150	5	3	Euclidean
M1 ECFP4 (GAULTON <i>et al.</i> , 2017)	769	1,025	2	Tanimoto
M1 ECFP6 (GAULTON <i>et al.</i> , 2017)	769	1,025	2	Tanimoto
Mfeat-factors (VANSCHOREN <i>et al.</i> , 2014)	2,000	216	10	Euclidean
Mfeat-Karhunen (VANSCHOREN <i>et al.</i> , 2014)	2,000	65	10	Euclidean
Seeds (LICHMAN, 2013)	210	8	3	Euclidean
Segmentation (VANSCHOREN <i>et al.</i> , 2014)	2,100	20	7	Euclidean
Semeion (VANSCHOREN <i>et al.</i> , 2014)	1,593	256	10	Cosine
Stock (VANSCHOREN <i>et al.</i> , 2014)	950	10	2	Euclidean
Transplant (VANSCHOREN <i>et al.</i> , 2014)	131	4	2	Euclidean
WDBC (LICHMAN, 2013)	569	32	2	Euclidean
Wine (LICHMAN, 2013)	178	13	3	Euclidean
Yeast galactose (YEUNG; MEDVEDOVIC; BUMGARNER, 2003)	205	81	4	Euclidean

Source: Elaborated by the author.

in terms of computational efficiency. We describe the experimental setup which refers to the methods discussed in Chapters 3 and 5.

5.4.1 Performance measure

We report the macro-averaged F-measure, i.e., the average over all classes of the harmonic mean between precision and recall for each class (SOKOLOVA; LAPALME, 2009). We compute the F-measure based only on those objects whose labels have not been exposed to the semi-supervised classification algorithms for training.

5.4.2 Datasets

We use datasets with different characteristics (such as number of objects, number of attributes, and number of classes) and from different domains (biology, text, and broadly from the UCI machine learning repository (LICHMAN, 2013)), requiring different distance measures, as summarized in Table 3.

Some datasets required pre-processing. The datasets “ACE” (ACE ECFP4 and ACE ECFP6), “COX2 ECPF6”, “DHFR” (DHFR ECFP4 and DHFR ECFP6), “Fontaine” (Fontaine ECFP4 and Fontaine ECFP6) and “M1” (M1 ECFP4 and M1 ECFP6) describe molecules utilized in the process of identifying relationships between chemical structure and biological activity (RIVERA-BORROTO *et al.*, 2011). We transformed the data with *generatemd* (a tool in the JChem framework, available at <<http://www.chemaxon.com>>) into a set of binary attributes using two different configurations of the Extended-Connectivity Fingerprints (ECFP), with the maximum diameter of the circular neighbors considered for each atom set to 4 and 6, resulting in ECFP4 and ECFP6, respectively. We applied the Tanimoto dissimilarity to perform the experiments in the new set of attributes. For the datasets “Auto price”, “Bank note–Authentication”, “Stock”, and “Transplant” we used the Euclidean distance on the *z*-score normalized data objects, which relates to Pearson correlation in the original data space.

5.4.3 Pre-labeled data subsets

For the semi-supervision, we selected labeled objects from the datasets randomly, ensuring that there is at least one label from each class, repeating the random selection 30 times, thus resulting in 30 variants of each dataset for each percentage of labeled objects. To study the influence of the amount of labeled data, we take different percentages of labeled objects: 2%, 5%, 8%, and 10%.

5.4.4 Algorithms and parameters

The original implementation of HISSCLU has been provided by the authors. As further competitors we included three semi-supervised classification approaches: the Gaussian Field and Harmonic Function (GFHF) (ZHU; GHAHRAMANI; LAFFERTY, 2003), the Robust Multi-class Graph Transduction (RMGT) (LIU; CHANG, 2009), and the Laplacian Support Vector Machine (LapSVM) (BELKIN; NIYOGI; SINDHWANI, 2006), for which implementations are available from a former comparative study (SOUSA; REZENDE; BATISTA, 2013). Following the recommendations in that study, we construct a graph and weight matrix for the label propagation in these algorithms as follows (SOUSA; REZENDE; BATISTA, 2013):

1. Compute the distance matrix using the dissimilarity function listed in Table 3;

2. Construct the graph using a symmetric version of the k -nearest neighbors graph. The mutual k -nearest neighbors graph creates an edge between objects $\mathbf{x}_i$ and $\mathbf{x}_j$ if and only if they are one of the k closest neighbors of each other.
3. Compute the weight matrix applying the radial basis function kernel (RBF kernel):

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}\right),$$

where σ is the kernel bandwidth parameter.

We set the bandwidth parameter as $\sigma = \frac{\sum_{i=1}^n d(\mathbf{x}_i, \mathbf{x}_{i_k})}{3n}$, following [Sousa, Rezende and Batista \(2013\)](#), where $d(\mathbf{x}_i, \mathbf{x}_{i_k})$ denotes the distance between $\mathbf{x}_i$ and its k -th nearest neighbor $\mathbf{x}_{i_k}$.

For the label propagation with algorithms GFHF and LapSVM in datasets with multiple classes, we use the standard *one-vs-all* combination of the binary classifiers.

For HISSCLU, HDBSCAN*(cd,wPWD), HDBSCAN*(cd,wMST), HDBSCAN*(ap,wPWD), and HDBSCAN*(ap,wMST) we set the parameters of the weighting function (to improve the separation between classes) as $\rho = 50.0$ and $\xi = 5.0$, which were also used in the original HISSCLU publication ([BÖHM; PLANT, 2008](#)). For HDBSCAN*(ap,-), no parameter is required.

For the neighborhood size, we tested a smaller ($m_{\text{pts}} = k = 4$) and a larger choice ($m_{\text{pts}} = k = 15$). The density-based semi-supervised classification algorithms achieve better results on average with the smaller value (confirming observations in previous studies ([CAMPELLO et al., 2015](#); [LELIS; SANDER, 2009](#); [BÖHM; PLANT, 2008](#))), while GFHF, RMGT, and LapSVM show better performance on average with larger neighborhood size (also confirming previous findings ([SOUSA; REZENDE; BATISTA, 2013](#))). Hence, to not give an unfair advantage to a class of algorithms, we run each of the algorithms with the value that works better for them on average, $m_{\text{pts}} = 4$ for the density-based methods, and $k = 15$ for the other ones.

5.4.5 Statistical test for performance comparison

To analyze the overall results, we applied the two-step procedure described by ([DEMSAR, 2006](#)). First, we applied the Friedman test ([FRIEDMAN, 1937](#)) to examine whether there is a significant difference between the results of the algorithms on collected datasets. If the null hypothesis, assuming no significant difference between the algorithms, is rejected at the given p -value, the Nemenyi posthoc test ([NEMENYI, 1962](#)) is applied to reveal which pairs of algorithms exhibit such differences. This test states that the performance of two different algorithms is significantly different if the corresponding average ranks differ by at least a Critical Difference (CD) value. In both tests, Friedman test and Nemenyi posthoc procedure, we selected a significance level of 5% ($\alpha = 0.05$).

5.4.6 Runtime experiments

To assess the runtime behavior of the algorithms we used a dedicated node of a large cluster, running one Intel Xeon E5-2667v4 processor at 3.2GHz with 64GB of available RAM. The algorithms were implemented in Java, with exception of the algorithm LapSVM, implemented in Matlab (MATLAB..., 2011). The datasets for the runtime experiment were generated randomly using “animals.c” (LICHMAN, 2013). The program generates objects that have 72 attributes and that belong to one of 4 classes. We generated 20000 instances for this dataset, and extracted subsets of size 100, 500, 1000, 5000, 10000, and 15000 instances, with an approximately equal distribution of the classes. We select randomly 30 labeled objects, at least one for each class, repeat the procedure 20 times, and present the mean runtime. We use the same parameters as in the effectiveness experiments for all algorithms.

5.5 Results

In this section we describe and analyze the results of our experiments in the semi-supervised classification scenario.

5.5.1 Effectiveness

Figure 16 presents the distribution of the achieved quality over all datasets, separated by the percentage of labeled objects used for training. A straightforward and unsurprising observation is that all algorithms improve their performance with an increased amount of training data, and most show also a reduced variance of quality over the datasets. A noticeable exception to the reduced variance with larger amounts of training data is HDBSCAN*(ap,wPWD); its variance actually increases with more training data. A direct comparison with HDBSCAN*(ap,wMST) and with HDBSCAN*(cd,wPWD) suggests that the combination of the all-points core distance and the label-based distance weighting applied to the distance matrix becomes more susceptible to random effects in the selection of labeled objects.

We see in general, that all variants have their strengths and weaknesses over the different datasets and dataset variants. When establishing a ranking of the algorithms and framework variants over the combined results (all datasets and all percentages of labeled objects), the Friedman and the Nemenyi post hoc tests check for statistical differences between the algorithms. Figure 17 visualizes the average ranks of the algorithms along with the critical distance. We can identify two distinct groups without statistical difference between the algorithms at confidence level $\alpha = 0.05$.

The top group contains four density-based algorithms: HDBSCAN*(cd,wMST), HDBSCAN*(cd,wPWD), HDBSCAN*(ap,wMST), and HISSCLU, together with two traditional algorithms: RMGT and LapSVM. The critical distance indicates that the differences in ranking

Figure 16 – Quality of results obtained by the semi-supervised classification algorithms averaged over all datasets, separated by different percentage of labeled objects.

Source: Research data.

Figure 17 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test.

Source: Research data.

Figure 18 – Runtime over dataset size.

Source: Research data.

within this group are not significant. The lower group (consisting of two overlapping groups) comprises the HDBSCAN* variants without any label-based distance weighting, the variant HDBSCAN*(ap,wPWD) that we already observed as having a larger variance of quality, and the last of the traditional competitors, GFHF.

5.5.2 Efficiency

Figure 18 depicts the runtime behavior of the tested algorithms when increasing the dataset size. Although comparisons of absolute runtime measurements are in our favor here, they should not be considered conclusive, especially when using different implementations (KRIEGL; SCHUBERT; ZIMEK, 2017). However, we can in fact observe that the algorithms LapSVM, GFHF, and RMGT have a worse runtime *behavior* compared to the density-based algorithms (note the log-log-scale of the plot). Among the density-based algorithms, HISSCLU is clearly worse than all the variants of our framework. Among our variants, applying weights on the complete distance matrix is again worse, as expected. Note that applying the weighting

on the MST instead barely manifests itself in terms of runtime; compare $\text{HDBSCAN}^*(\text{cd}, -)$ vs. $\text{HDBSCAN}^*(\text{cd}, \text{wMST})$ and $\text{HDBSCAN}^*(\text{ap}, -)$ vs. $\text{HDBSCAN}^*(\text{ap}, \text{wMST})$: in both pairs the difference is hardly noticeable and is in the range of milliseconds. Applying weights on the distance matrix results a more substantial change. Overall, the runtime experiments support theoretical expectations.

5.6 Final remarks

In this chapter we present a completely new framework for semi-supervised classification. Such framework is based on density-based clustering algorithms, which were overlooked in the semi-supervised classification area. The novel framework is implemented using as core the HDBSCAN^* framework that has on its base a Minimum Spanning Tree built with the Mutual reachability distance as weight (MST_r). With the building blocks of the new framework it is possible to derive algorithms such as HISSCLU and SSDBSCAN, with the shortcomings identified during the review of these algorithms.

The new framework showed to be efficient as algorithms such as GFHF, RMGT and LapSVM. It also presents possible good quality results than the aforementioned algorithms. In the next Chapter we present another contribution of this document. The next contribution also extends HDBSCAN^* algorithm, now on the semi-supervised clustering scenario.

LABEL BASED SEMI-SUPERVISED FOSC

When performing semi-supervised *classification*, all classes are known in advance, pre-labeled objects from all of these classes are available, and all unlabeled objects are in principle supposed to be labeled by the algorithm. In contrast, when the task at hand is semi-supervised *clustering*, not all categories are necessarily known in advance, which means that labels may not be available for some (unknown) classes yet to be discovered, and part of the unlabeled objects may be left unclustered as noise. In this case, the framework proposed in [Chapter 5](#) is no longer suitable.

HDBSCAN* offers an optional post-processing method of its clustering hierarchy, called FOSC (described in [section 2.4](#)), that can extract a flat clustering solution by performing local cuts through the hierarchy in order to select a collection of non-overlapping clusters that is optimal according to a given unsupervised or semi-supervised criterion.

In this Chapter, we introduce a new semi-supervised quality measure for the optimal cluster extraction procedure (FOSC) used by HDBSCAN* that operates directly with labels rather than constraints. We also describe how this measure can be combined with the unsupervised measure of cluster *Stability* (see [section 2.4](#)) for optimal cluster extraction, making the resulting combination effective irrespective of whether only part, all or none of the clusters in the data are represented by labeled observations.

6.1 Label-based quality criteria for optimal cluster extraction

Our newly proposed semi-supervised measure of cluster quality is based on the B^3 *Precision* and B^3 *Recall* criteria originally proposed by [Bagga and Baldwin \(1998\)](#) and subsequently studied by [Amigó et al. \(2009\)](#) in the context of an external cluster validation index, called B^3 (BCubed). These criteria take pairs of objects into account, but they are computed individually

Figure 19 – B^3 Precision and B^3 Recall of an object x_j . Objects within the ellipse belong to the same cluster. Red crosses and blue stars represent two different class labels. Unlabeled objects are represented by black filled circles. Notice that any two objects with the same class label in the same cluster have the same precision and recall.

Source: Elaborated by the author.

for each pre-labeled object $\mathbf{x} \in \mathbf{X}_L$. Specifically, the B^3 Precision of $\mathbf{x}$ measures the proportion of pre-labeled objects in the same cluster as $\mathbf{x}$ that share the same class label as $\mathbf{x}$, including $\mathbf{x}$ itself. B^3 Recall measures the proportion of objects with the same class label as $\mathbf{x}$ sharing the same cluster with $\mathbf{x}$.

To compute B^3 Precision and B^3 Recall we consider only the set of pre-labeled objects. Given an object $\mathbf{x} \in \mathbf{X}_L$ with class label $\text{class}(\mathbf{x})$ and a cluster $\mathbf{C}_i$ containing this object, the B^3 Precision of $\mathbf{x} \in \mathbf{C}_i$ can be formally defined as:

$$P_{B^3}(\mathbf{x}, \mathbf{C}_i) = \frac{|\{\mathbf{x}' \mid \mathbf{x}' \in \{\mathbf{C}_i \cap \mathbf{X}_L\} \wedge \text{class}(\mathbf{x}) = \text{class}(\mathbf{x}')\}|}{|\{\mathbf{x}' \mid \mathbf{x}' \in \{\mathbf{C}_i \cap \mathbf{X}_L\}\}|} \quad (6.1)$$

and the B^3 Recall of $\mathbf{x} \in \mathbf{C}_i$ can be defined as:

$$R_{B^3}(\mathbf{x}, \mathbf{C}_i) = \frac{|\{\mathbf{x}' \mid \mathbf{x}' \in \{\mathbf{C}_i \cap \mathbf{X}_L\} \wedge \text{class}(\mathbf{x}) = \text{class}(\mathbf{x}')\}|}{|\{\mathbf{x}' \mid \mathbf{x}' \in \mathbf{X}_L \wedge \text{class}(\mathbf{x}) = \text{class}(\mathbf{x}')\}|} \quad (6.2)$$

Figure 19 illustrates these concepts for a pre-labeled object x_j as part of a cluster with eleven objects, seven of them pre-labeled, four of which share the same label as x_j . Obviously any two objects with the same class label in the same cluster share the very same values of P_{B^3} and R_{B^3} , which allows fast computation.

As in traditional assessment of supervised classifiers, precision and recall capture two different aspects of an outcome, one of which will be overlooked if a single criterion is chosen. Just like the well-known F1-Measure in classification, a single conservative index (hereafter called B^3 F-Measure) can be obtained by combining P_{B^3} and R_{B^3} taking their harmonic mean:

$$F_{B^3}(\mathbf{x}, \mathbf{C}_i) = \frac{2P_{B^3}(\mathbf{x}, \mathbf{C}_i) \cdot R_{B^3}(\mathbf{x}, \mathbf{C}_i)}{P_{B^3}(\mathbf{x}, \mathbf{C}_i) + R_{B^3}(\mathbf{x}, \mathbf{C}_i)} \quad (6.3)$$

The B^3 F-Measure in Equation (6.3) can be used as a building block for an optimization criterion to extract clusters from the HDBSCAN* hierarchy using FOSC. Specifically, let $\{C_1, \dots, C_k\}$ be the set of all candidate clusters in a HDBSCAN* hierarchy of $\mathbf{X}$ from which we want to extract a flat clustering solution in a semi-supervised way. The goal is to maximize the *Overall B^3 F-Measure* of the resulting solution, which is an average B^3 F-Measure over the pre-labeled objects as belonging to their respective selected clusters:

$$Overall_{F_{B^3}} = \frac{1}{|\mathbf{X}_L|} \sum_{i=1}^k \left(\sum_{\mathbf{x} \in \{C_i \cap \mathbf{X}_L\}} \delta_i \cdot F_{B^3}(\mathbf{x}, C_i) \right) \quad (6.4)$$

where δ_i is the indicator function used by FOSC to determine whether or not ($\delta_i = 1$ or $\delta_i = 0$, respectively) cluster C_i is selected to be part of the optimal flat solution.

Notice that the Overall B^3 F-Measure in Equation (6.4) satisfies the properties of *additivity* and *locality* required by FOSC. In fact, it can be easily decomposed as:

$$Overall_{F_{B^3}} = \sum_{i=1}^k \delta_i \cdot \omega(C_i) \quad (6.5)$$

where

$$\omega(C_i) = \frac{1}{|\mathbf{X}_L|} \left(\sum_{\mathbf{x} \in \{C_i \cap \mathbf{X}_L\}} F_{B^3}(\mathbf{x}, C_i) \right) \quad (6.6)$$

The Overall B^3 F-Measure is therefore a sum of individual components $\omega(C_i)$ that can be pre-computed independently for each candidate cluster in the cluster tree as it depends solely on the information of the pre-labeled objects that belong to that cluster.

One could analogously define an *Overall B^3 Precision* and *Overall B^3 Recall* if desired, which could also be decomposed in the same way. We omit the details here as they follow closely and straightforwardly the development shown above.

Figures 20b, 20c, and 20d show the cluster tree for the hierarchy in Table 4 with the quality value of each cluster individually assessed by the decomposed components of the *Overall B^3 Precision*, *Overall B^3 Recall*, and *Overall B^3 F-Measure* as described above, respectively, for a collection of 5 pre-labeled objects, $\mathbf{X}_L = \{\mathbf{x}_1, \mathbf{x}_6, \mathbf{x}_8, \mathbf{x}_{15}, \mathbf{x}_{18}\}$, with 4 different class labels (red cross, green hash, magenta triangle, and blue star).

The B^3 Precision, B^3 Recall, and B^3 F-Measure for singleton nodes corresponding to noise objects that are not pre-labeled (omitted for the sake of clarity) are undefined, and they are set to zero so that these nodes do not affect computations. The singleton node descendant from C_2 containing $\mathbf{x}_1$, which is pre-labeled (red cross), does affect computations in all semi-supervised scenarios, as shown in Figures 20b–20d.

Figure 20b shows that B^3 Precision has guided FOSC to extract the leaf nodes, as the corresponding clusters tend to be purer in class labels. In contrast, B^3 Recall has guided FOSC

Figure 20 – HDBSCAN* cluster tree for the hierarchy in Table 4 and flat clustering extraction using FOSC: (a) Unsupervised extraction using Stability; (b) — (d) Labeled-based semi-supervised extraction using B^3 Precision, Recall, and F-Measure, respectively; (e) Mixed case. The set of pre-labeled objects, $\mathbf{X}_L$, is represented using colored symbols: $\mathbf{x}_1$ and $\mathbf{x}_6$ as red crosses, $\mathbf{x}_8$ as a green hash, $\mathbf{x}_{15}$ as a magenta triangle, and $\mathbf{x}_{18}$ as a blue star. Singleton nodes corresponding to noise objects that are not pre-labeled are omitted as they do not affect computations. Extracted clusters in each case are highlighted in bold.

Source: Elaborated by the author.

Table 4 – Cluster hierarchy, extracted from the illustrative dataset presented in Figure 2, with $m_{\text{pts}} = m_{\text{CISize}} = 3$. Colors (red, blue, green, and magenta) highlight a collection of pre-labeled objects, $\mathbf{X}_L = \{\mathbf{x}_1, \mathbf{x}_6, \mathbf{x}_8, \mathbf{x}_{15}, \mathbf{x}_{18}\}$, which can be used to extract clusters from the hierarchy in a semi-supervised way (cf. Figure 20).

ε	$\mathbf{x}_1$	$\mathbf{x}_5$	$\mathbf{x}_{20}$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_{13}$	$\mathbf{x}_{14}$	$\mathbf{x}_{12}$	$\mathbf{x}_{15}$	$\mathbf{x}_{16}$	$\mathbf{x}_4$	$\mathbf{x}_{22}$	$\mathbf{x}_{17}$	$\mathbf{x}_{18}$	$\mathbf{x}_{19}$	$\mathbf{x}_{21}$	$\mathbf{x}_6$	$\mathbf{x}_9$	$\mathbf{x}_{11}$	$\mathbf{x}_7$	$\mathbf{x}_8$	$\mathbf{x}_{10}$
3.30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3.19	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
3.18	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.24	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.22	0	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3
1.17	0	0	0	0	4	4	4	4	4	4	5	5	5	5	5	5	3	3	3	3	3	3
1.00	0	0	0	0	4	4	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.98	0	0	0	0	0	4	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.85	0	0	0	0	0	0	4	4	4	4	0	5	5	5	5	5	3	3	3	3	3	3
0.82	0	0	0	0	0	0	4	4	4	4	0	0	5	5	5	5	3	3	3	3	3	3
0.72	0	0	0	0	0	0	0	4	4	4	0	0	5	5	5	5	3	3	3	3	3	3
0.63	0	0	0	0	0	0	0	4	4	4	0	0	0	5	5	5	3	3	3	3	3	3
0.61	0	0	0	0	0	0	0	4	4	4	0	0	0	5	5	5	6	6	6	7	7	7
0.60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6	6	7	7	7
0.51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6	6	0	0	0
0.00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Source: Elaborated by the author.

to extract the root (Figure 20c).¹ This is not necessarily always the case, in particular because there may be ties between the values of precision or recall of a cluster and that of its sub-clusters. In the semi-supervised setting, the way FOSC resolves ties in the values of the objective function involving different candidate solutions in a subtree is by taking the solution that would be chosen in the unsupervised setting (CAMPELLO *et al.*, 2013). In HDBSCAN*, this means that ties in the semi-supervised cluster extraction are decided using Stability (CAMPELLO *et al.*, 2015). Notice that this is particularly important when certain regions of the data, corresponding to entire subtrees of the cluster tree, are not represented by labels at all, so decisions inside those subtrees can only be made in an unsupervised way.

The B³ F-Measure provides a conservative (pessimistic) compromise between precision and recall, which in this particular example has led FOSC to extract the same solution as B³ Precision (see Figure 20d). In general, however, the solution extracted with B³ F-Measure does not need to coincide with either B³ Precision or B³ Recall.

FOSC also allows cluster extraction by using an objective function that combines the unsupervised and semi-supervised measures into a single function. In particular, it has been shown in the original FOSC publication (CAMPELLO *et al.*, 2015) that any convex combination of objective functions that satisfy the properties of additivity and locality required by the algorithm will also satisfy the property of additivity and locality, and its decomposition into local

¹ The root can be trivially removed from the set of candidate solutions in FOSC, if desired (CAMPELLO *et al.*, 2013).

components corresponding to each individual cluster is given by the convex combination of the corresponding local components of the original objective functions.

Here, we experiment with the average between Overall Stability (unsupervised) and Overall B^3 F-Measure (semi-supervised). Since the scales of these different objective functions are different, they have to be normalized first. The B^3 F-Measure ranges within $[0, 1]$, but Stability has no upper bound. In order to make the mixed measure commensurable in this case, the Stability of individual clusters can be divided by the overall value of Stability of the optimal flat solution that is extracted in the unsupervised case (CAMPELLO *et al.*, 2013). For example, recall that the optimal unsupervised solution in Figure 20a has an Overall Stability of 14.95. We then divide the Stability of each cluster by this value before taking the average with the corresponding value of F-Measure. In our example, the result is shown in Figure 20e.

Notice in Figure 20e that, by combining both unsupervised and semi-supervised measures, FOSC has been able to extract a solution that is different from either case ($\mathbf{P} = \{\mathbf{C}_4, \mathbf{C}_5, \mathbf{C}_3\}$). In particular, since the Stability of $\mathbf{C}_2$ is not much higher than that of its sub-clusters combined, the presence of pre-labeled objects with different labels in that cluster has driven FOSC to choose the sub-clusters instead. Contrarily, in spite of the presence of pre-labeled objects with different labels in $\mathbf{C}_3$, FOSC has opted to keep this cluster rather than splitting it, because Stability strongly suggests that this is a single cluster. Of course, if full priority is to be given to user-defined labels, there is no point in using the mixed approach, and B^3 F-Measure should be used instead (Stability being only used to decide ties).

HDBSCAN* and FOSC take the so-called *soft* approach to semi-supervised clustering (BASU; DAVIDSON; WAGSTAFF, 2008), in which labels or constraints are (prior) user expectations rather than hard constraints that must be enforced. The particular approach can be interpreted as a strategy that gives priority to satisfy the implicit model assumptions (density connectivity in the case of HDBSCAN*) when constructing the cluster hierarchy, and then use external information provided by the user as preferences (rather than hard requirements) to extract a flat solution (CAMPELLO *et al.*, 2013; CAMPELLO *et al.*, 2015).

6.2 Complexity

For a given cluster tree containing k candidate clusters, each of which has an associated quality measure that has been pre-computed, FOSC can be implemented in a very efficient way with two traversals through the cluster tree, one bottom-up as previously described and another one top-down just materializing the provisional selections. This means that the complexity of the algorithm is $O(k)$, i.e., linear w.r.t. the number of nodes in the tree, both in terms of running time and memory space (CAMPELLO *et al.*, 2013). In simplified cluster trees such as those produced by HDBSCAN*, k is typically much smaller than the number of data objects ($k \ll n$). Even in an unlikely scenario where a binary cluster split is observed at each of the n (maximum) possible

hierarchical levels, it follows that $k = 2n - 1$ and, therefore, FOSC is $O(n)$ in the worst-case (given a cluster tree and the corresponding values of cluster quality).

The unsupervised measure of Stability can be computed by HDBSCAN* “on-the-fly”, i.e., as the hierarchical levels are iteratively computed, so Stability does not increase the computational complexity of HDBSCAN*, which is $O(n^2)$ w.r.t. runtime,² except for a small constant factor. Likewise, it should be clear that B^3 Precision, B^3 Recall, and B^3 F-Measure can be trivially computed for all candidate clusters simultaneously to the construction of the hierarchy, by just keeping track of the number of pre-labeled objects of each class in each cluster, which again can be done without affecting the computational complexity of HDBSCAN*. Even in the scenario where B^3 Precision, B^3 Recall, and B^3 F-Measure are computed afterwards, as a post-processing of the HDBSCAN* hierarchy, it is straightforward to compute these measures for each cluster with a single pass through the hierarchical levels for each pre-labeled object, updating the counts of the respective class label at each cluster the object belongs to in the cluster tree. In this case, the additional post-processing cost in the worst-case, where the clustering hierarchy has n levels, is $O(n \cdot |\mathbf{X}_L|)$, which again, does not change the computational complexity of HDBSCAN* since $|\mathbf{X}_L| < n$ (typically $|\mathbf{X}_L| \ll n$).

6.3 Experimental setup

In this section we present the experimental setup for the semi-supervised clustering scenario, such as performance measure, datasets selection of the labeled objects and the statistical tests.

6.3.1 Performance measures

In the semi-supervised clustering scenario, we report the results of our experiments using the Adjusted Rand Index (ARI) (HUBERT; ARABIE, 1985), which is a standard external validation measure in the clustering literature (JAIN; DUBES, 1988). The index has been computed here *not* using any of the objects belonging to the set of pre-labeled objects. Noise objects are treated as singletons for the ARI computations.

6.3.2 Datasets

For controlled experiments where we know the true probability distribution of clusters, we use the benchmark collection of Handl and Knowles (2007),³ which contains 160 synthetic datasets with Gaussian and Ellipsoidal clusters. There are two sub-collections, one containing 80 low dimensional datasets (2D and 10D) and the other containing 80 high dimensional datasets (50D and 100D). The datasets have 4, 10, 20, and 40 clusters with cluster sizes varying from 10

² It can be implemented in $O(n)$ w.r.t. memory (CAMPELLO *et al.*, 2015).

³ Available at <http://personalpages.manchester.ac.uk/staff/Julia.Handl/generators.html>

Table 5 – List of datasets collected to perform the semi-supervised clustering experiments.

Dataset		#obj	#att	#cl	Distance
Real					
Articles-1442-5	(NALDI <i>et al.</i> , 2011)	253	4636	5	Cosine
Articles-1442-80	(NALDI <i>et al.</i> , 2011)	253	388	5	Cosine
Bank note–Authentication	(VANSCHOREN <i>et al.</i> , 2014)	1,372	5	2	Euclidean
Cardiotocography	(VANSCHOREN <i>et al.</i> , 2014)	2,126	36	10	Euclidean
CellCycle-237	(YEUNG <i>et al.</i> , 2001)	237	17	4	Euclidean
Chowdary	(SOUTO <i>et al.</i> , 2008)	104	183	2	Cosine
Diggle table	(VANSCHOREN <i>et al.</i> , 2014)	310	8	9	Euclidean
Ecoli	(LICHMAN, 2013)	336	7	8	Euclidean
Gordon	(SOUTO <i>et al.</i> , 2008)	181	1,627	2	Cosine
Iris	(LICHMAN, 2013)	150	5	3	Euclidean
Mfeat-factors	(VANSCHOREN <i>et al.</i> , 2014)	2,000	216	10	Euclidean
Mfeat-Karhunen	(VANSCHOREN <i>et al.</i> , 2014)	2,000	65	10	Euclidean
Seeds	(LICHMAN, 2013)	210	8	3	Euclidean
Segmentation	(VANSCHOREN <i>et al.</i> , 2014)	2,100	20	7	Euclidean
Stock	(VANSCHOREN <i>et al.</i> , 2014)	950	10	2	Euclidean
WDBC	(LICHMAN, 2013)	569	32	2	Euclidean
Wine	(LICHMAN, 2013)	178	13	3	Euclidean
Yeast galactose	(YEUNG; MEDVEDOVIC; BUMGARNER, 2003)	205	81	4	Euclidean
ALOI collections					
ALOI PCA	(HORTA; CAMPELLO, 2012)	[50, 125]	6	[2, 5]	Euclidean
ALOI 88	(HORTA; CAMPELLO, 2012)	[50, 125]	88	[2, 5]	Euclidean
Artificial collections					
Gaussian	(HANDL; KNOWLES, 2007)	[200, 5000]	[2, 10]	[4, 40]	Euclidean
Ellipsoid	(HANDL; KNOWLES, 2007)	[200, 5000]	[50, 100]	[4, 40]	Euclidean

Source: Elaborated by the author.

to 500 depending on the number of clusters. For each combination of dimensionality and number of clusters there are 10 different datasets. We used Euclidean distance with this collection.

We also use real datasets from different domains. In particular, part of the datasets considered in the semi-supervised classification scenario (section 5.4) are reused here in the semi-supervised clustering scenario. However, since classes as represented by classification labels do not necessarily correspond to clusters from a density-based perspective, we select only a subset of the datasets in Table 3 for which at least one clustering algorithm (amongst our proposed methods or among our baseline competitors) is able to achieve a solution with ARI of at least 0.50. This way, we ensure that the comparisons involve datasets for which it is possible to recover at least partially a clustering structure. The selected datasets are listed in Table 5.

For the sake of completeness, we also include an additional dataset collection as well as additional individual datasets that have been used for clustering experiments in the HDBSCAN*

paper (CAMPELLO *et al.*, 2015), some of which are not included in Table 3. Specifically, datasets “Articles-1442-5” and “Articles-1442-80” consist of high dimensional (Bag-of-Words) representations of text documents, originally used by Naldi *et al.* (2011). These datasets are formed by 253 articles from 5 categories each, represented by 4636 and 388 dimensions, respectively. We used Cosine similarity for these datasets, as common in text analysis. Dataset “CellCycle-237” (YEUNG *et al.*, 2001) contains the expression levels of 237 genes, belonging to 4 known categories, across 17 conditions/dimensions. For this dataset we used Euclidean distance with z -score normalization of objects, which relates to Pearson correlation in the original data space (the standard measure in gene-expression analysis). Dataset “Ecoli” is from the UCI Repository (LICHMAN, 2013), and contains 336 objects and 7 dimensions, with 8 classes. For this dataset we used Euclidean distance.

Finally, the “ALOI” collections in Table 5 consist of real datasets composed of image features of images extracted from the Amsterdam Library of Object Images (ALOI) (GEUSE-BROEK; BURGHOUTS; SMEULDERS, 2005) and processed as described by Horta and Campello (2012). These datasets were created by randomly selecting c ALOI image categories as class (cluster) labels, 100 times for each $c = 2, 3, 4, 5$, then sampling (without replacement), each time, 25 images from each of the c selected categories, resulting in 400 sets, each of which contains 2, 3, 4, or 5 classes (clusters) and 50, 75, 100, or 125 images (objects). The images were represented using 6 different image descriptors, with 144, 88, 128, 5, 44, and 256 attributes, respectively. The datasets using the *texture statistics* descriptor are denoted by “ALOI-TS88”, whereas “ALOI-PCA” stands for the datasets with a 6-dimensional representation combining the first principal component extracted from each of the six descriptors, using PCA. For both configurations, we used Euclidean distance.

Note that the “real” datasets are individual datasets, while ALOI and Artificial are sets of datasets. We will show summary results for each of the three categories (i.e., the collection of “real” datasets, the ALOI collections, and the artificial collections) separately.

6.3.3 Pre-labeled data subsets

In order to obtain the subset of pre-labeled objects, $\mathbf{X}_L$, we randomly selected labeled objects from the datasets following two different strategies: (a) *non-controlled random*, where we draw objects without replacement, but under no further constraints; and (b) *controlled random*, where we ensure that a certain fraction of the class labels will (or will not) be missing in $\mathbf{X}_L$. In the first, non-controlled random setting, it is *possible* that certain class labels will not be represented in the resulting subset of pre-labeled objects, which represents applications of semi-supervised clustering involving unknown categories yet to be discovered. In this setting, we varied the percentage of pre-labeled objects as 0% (unsupervised case, as a baseline), 1%, 2% and 5% of the data. For the second, controlled random setting, we enforce that a fraction of the c class labels in the ground truth will be missing in $\mathbf{X}_L$. We experiment and compare the results

with no classes missing (by ensuring that at least one object from each class is selected), $\lceil \frac{c}{2} \rceil$ classes missing, and all c classes missing (unsupervised case). For the first two cases, we varied the percentage of pre-labeled objects as 1%, 2% and 5% of the data. The results reported are averages over 50 random selections of the pre-labeled objects, for each dataset.

Some of the algorithms used as baseline competitors in our experiments perform semi-supervised clustering based on instance-level constraints, rather than on labels. For these algorithms, we also produce a corresponding set of instance-level constraints from each subset $\mathbf{X}_L$ of pre-labeled objects randomly drawn in our experiments; all possible pairwise constraints that hold in $\mathbf{X}_L$ are generated as follows: (a) a should-link constraint is created for each pair of objects with the same label in $\mathbf{X}_L$; and (b) a should-not-link constraint is created for each pair of objects with different labels in $\mathbf{X}_L$. Note that using instance-level constraints requires special care for a fair evaluation setup to not use information in the evaluation that has been implicitly exposed to the learning procedure (POURRAJABI *et al.*, 2014). We performed the evaluation accordingly.

6.3.4 Algorithms and Parameters

We include again HISSCLU as a competitor using the code provided by the authors, but now making use of the clustering extraction procedure described as a post-processing routine in the original publication (BÖHM; PLANT, 2008), called *k-cluster* extraction (see subsection 3.2.3). We also include as competitor SSDBSCAN, using the original code provided by *Lelis and Sander* (2009). We compare their original code against our improved code, hereafter referred to as SSDBSCAN++, which takes advantage of our density-based framework and runs the algorithm on top of a single, pre-computed MST_r , in contrast to the original algorithm, which dynamically and partially builds minimum spanning trees multiple times. As for the core algorithm in our framework, HDBSCAN*, we compare our new clustering extraction strategies with the strategies from the original publication (CAMPELLO *et al.*, 2015), namely: unsupervised FOSC extraction based on *stability*, semi-supervised FOSC extraction based on instance-level pairwise constraints (unsupervised stability used to decide ties only), and the mixed case involving a balanced combination of these two strategies. We will call these competitors HDBSCAN*(UN), HDBSCAN*(CON), and HDBSCAN*(MixCON), respectively. Our proposed, label-based counterparts to HDBSCAN*(CON) and HDBSCAN*(MixCON) are called hereafter HDBSCAN*(BC) and HDBSCAN*(MixBC), respectively. The former performs semi-supervised FOSC extraction using the B^3 F-Measure, with the unsupervised criterion of stability being used to decide ties only; the latter is the mixed case involving a balanced combination of these two criteria, as discussed in section 6.1.

All these algorithms share as a common parameter the neighborhood size, m_{pts} , which has been set in the same way as previously described for the semi-supervised classification experiments (section 5.4). As for the additional parameters required by HISSCLU, ρ and ξ , the setting previously adopted for the semi-supervised classification experiments, which was

particularly effective in stretching class boundaries in that scenario, has shown not to be as effective in the clustering scenario. For this reason, we performed preliminary experiments with several combinations of these two parameters within the ranges $\rho \in [1, 50]$ and $\xi \in [0.5, 5.0]$, and selected $\rho = 10.0$ and $\xi = 5.0$ as the most effective choice for the clustering experiments. The only new parameter used here, the value of k in HISSCLU’s k -cluster extraction routine, was set as $k = 0.2$, which was also used by the authors in the original publication (BÖHM; PLANT, 2008).

6.3.5 Statistical test and runtime experiments

For performance comparisons in the clustering scenario, we apply the same statistical validation procedure as described in subsection 5.4.5 for the classification scenario. Runtime experiments follow the same experimental design as described in subsection 5.4.6.

6.4 Results

6.4.1 Improved SSDBSCAN

We start by comparing the results of our improved implementation of SSDBSCAN, called SSDBSCAN++, against the results of the original code provided by (LELIS; SANDER, 2009). The results averaged over each of the dataset collections in Table 5 are summarized in Table 6, for the non-controlled random scenario. The results in Table 6 show that, by using a single pre-computed MST_r , our improved implementation achieves at least the same quality of results as the original implementation, if not better (as highlighted in bold).

For the sake of compactness, we omit the results of the controlled random scenario, but the conclusions are the same. For this reason, we will include only SSDBSCAN++ in the subsequent experiments.

6.4.2 Effectiveness

Figure 21 shows the summarized results of the compared algorithms for the “Artificial” collection, under the non-controlled random setup, separated by the percentage of labeled objects used for training. The first column (0%) corresponds to the unsupervised case, which emphasizes two facts: (a) neither SSDBSCAN nor HISSCLU’s k -cluster can operate in the absence of pre-labeled objects; and (b) in this scenario, all HDBSCAN* semi-supervised variants, which resolve ties in an unsupervised way, reduce to the unsupervised case (i.e., HDBSCAN*(UN)). As we move across the other columns from 1% to 5%, it is clear that all semi-supervised algorithms benefit from larger amounts of pre-labeled objects. However, unlike the semi-supervised variants of HDBSCAN*, which operate reliably all across the board, SSDBSCAN and k -cluster are unstable and provide competitive results only when using 5% of pre-labeled objects. When

Table 6 – ARI results (mean $\pm$ standard deviation) of the original SSDBSCAN implementation and our improved implementation, SSDBSCAN++, in the non-controlled random scenario with different amounts of pre-labeled objects.

Collection	SSDBSCAN	SSDBSCAN++
1% pre-labeled obj.		
Real	0.37 ± 0.08	0.39 ± 0.09
ALOI	0.05 ± 0.01	0.05 ± 0.01
Artificial	0.63 ± 0.01	0.66 ± 0.01
2% pre-labeled obj.		
Real	0.45 ± 0.09	0.49 ± 0.09
ALOI	0.22 ± 0.01	0.22 ± 0.01
Artificial	0.73 ± 0.01	0.77 ± 0.01
5% pre-labeled obj.		
Real	0.53 ± 0.08	0.59 ± 0.07
ALOI	0.58 ± 0.01	0.59 ± 0.01
Artificial	0.80 ± 0.02	0.84 ± 0.01

Source: Research data.

comparing our proposed methods, HDBSCAN*(BC) and HDBSCAN*(MixBC), against their constraint-based counterparts, HDBSCAN*(CON) and HDBSCAN*(MixCON), respectively, it is clear that, while HDBSCAN*(BC) and HDBSCAN*(CON) produce very similar results, HDBSCAN*(MixBC) outperforms HDBSCAN*(MixCON) in all scenarios.

A possible reason for SSDBSCAN and HISSCLU's k -cluster being unstable and producing poor results for smaller amounts of pre-labeled objects is that, in the non-controlled random experimental setup, there is no guarantee that all class labels are represented in the subset of pre-labeled objects, $\mathbf{X}_L$. To investigate this hypothesis, in Figure 22 we show the results for the controlled random setup, where we contrast experiments where all c class labels in the ground truth are guaranteed to be represented in $\mathbf{X}_L$, against experiments with the same amount of pre-labeled objects but only approximately half of the c class labels allowed to be in $\mathbf{X}_L$. Notice that the extreme case in which no class label in the ground truth is represented in $\mathbf{X}_L$ corresponds to the unsupervised case, shown in the first column of Figure 21.

Figure 22 confirms that, as expected, all algorithms are to some extent negatively affected by class labels missing in $\mathbf{X}_L$. However, while all the HDBSCAN* variants, including our proposed methods, are only slightly impacted, showing to be robust to this important factor in practical clustering applications, SSDBSCAN and k -cluster are strongly affected and show prominent drops in performance. Among the HDBSCAN* variants, it is noticeable that those variants that use the unsupervised criterion to decide ties only (i.e., HDBSCAN*(CON) and our HDBSCAN*(BC)), are slightly more sensitive to missing labels than the mixed vari-

Figure 21 – ARI results for the Artificial collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

ants, HDBSCAN*(MixCON) and our HDBSCAN*(MixBC), which instead combine the semi-supervised and the unsupervised criteria. For this reason, while HDBSCAN*(CON) and our HDBSCAN*(BC) perform slightly better than HDBSCAN*(MixCON) and our HDBSCAN*(MixBC) when no label is unrepresented, the opposite occurs when there are unrepresented labels.

The results for the “ALOI” collection, under the non-controlled random experimental setup, are shown in Figure 23. The conclusions that can be drawn are similar to those from the artificial collection (Figure 21). The differences are: (a) semi-supervision improves only slightly the results in the ALOI datasets. In fact, the difference between the semi-supervised and the unsupervised results is only noticeable when 5% of pre-labeled objects are used; (b) there is a smaller variability in the average results across different datasets (error bars are almost indistinguishable); (c) SSDBSCAN and k -cluster have now underperformed in all scenarios, even when 5% of pre-labeled objects are used; and (d) this time, the constraint-based mixed variant, HDBSCAN*(MixCON), has also underperformed when only 1% or 2% of pre-labeled objects have been used.

Figure 22 – ARI results for the Artificial collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

Figure 23 – ARI results for the ALOI collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

In order to investigate the cause for the underperformance of SSDBSCAN, k -cluster, and HDBSCAN*(MixCON), we show the results for the controlled random experiments in Figure 24. While it is clear that the methods perform similarly when all class labels are represented in $\mathbf{X}_L$, with SSDBSCAN outstanding to some extent, the performances of SSDBSCAN, k -cluster, and HDBSCAN*(MixCON) (the latter w.r.t. the 1% and 2% scenarios only) drop substantially when there are categories not represented in $\mathbf{X}_L$.

The results for the “Real” collection, under the non-controlled random experimental setup, are shown in Figure 25. In this case, apart from the higher variability in the results (due to summarizing over very different datasets), the conclusions are essentially the same as those drawn from the artificial collection in Figure 21. In particular, all semi-supervised algorithms benefit from larger amounts of pre-labeled objects, but while SSDBSCAN and k -cluster show superior results when using larger amounts of pre-labeled objects (5%), which increases the chances that all or most of the class labels are represented in $\mathbf{X}_L$, these two algorithms underperform when smaller amounts are used (1%). When comparing our label-based methods,

Figure 24 – ARI results for the ALOI collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

Figure 25 – ARI results for the Real collection (non-controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

HDBSCAN*(BC) and HDBSCAN*(MixBC), against their constraint-based counterparts, HDBSCAN*(CON) and HDBSCAN*(MixCON), respectively, the results are very similar, but slightly higher for the label-based variants. When comparing the mixed variants, HDBSCAN*(MixBC) and HDBSCAN*(MixCON), with their non-mixed counterparts, HDBSCAN*(BC) and HDBSCAN*(CON), respectively, the non-mixed variants, which use unsupervised evaluation only to decide ties, exhibit better results. However, looking at the results for the controlled random setup, in Figure 26, it is clear that, similar to the artificial collection in Figure 22, the negative impact of unrepresented labels is more noticeable for the non-mixed variants. Their drop in performance when changing from c (100%) to $\lceil c/2 \rceil$ (50%) represented labels is only less prominent than SSDBSCAN's and k -cluster's.

When establishing a ranking of the algorithms over the combined results of each collection separately (all datasets within the collection, all percentages of labeled objects, and 50 random subsets of pre-labeled objects for each combination of these), the Friedman and the Nemenyi post hoc tests check for statistical differences between the algorithms. Figures 27, 28,

Figure 26 – ARI results for the Real collection (controlled random scenario). Error bars denote 95% confidence intervals for the mean of means within each dataset.

Source: Research data.

Figure 27 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: Artificial collection (non-controlled random scenario).

Source: Research data.

Figure 28 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: ALOI collection (non-controlled random scenario).

Source: Research data.

Figure 29 – Average ranks and critical distance (CD) with statistical significance $\alpha = 0.05$ according to the Friedman test: Real collection (non-controlled random scenario).

Source: Research data.

and 29 visualize the average ranks of the algorithms along with the critical distance of the test, for the collections Artificial, ALOI, and Real, respectively.

Some general conclusions based on the ranks are: (a) our proposed mixed variant, HDBSCAN*(MixBC), is the best ranked overall, in all three collections, with statistical difference to all other algorithms in the ALOI collection and to all algorithms but HDBSCAN*(CON) in the Artificial collection; and (b) when comparing our proposed HDBSCAN*(BC) with its constraint-based non-mixed counterpart, HDBSCAN*(CON), the latter is better ranked in the collections Artificial and ALOI, whereas the former is better ranked in the collection Real, but there is no statistical difference between these two variants in any of these cases.

Figure 30 – Runtime over dataset size.

Source: Research data.

6.4.3 Efficiency

Figure 30 depicts the runtime behavior of the tested algorithms. While SSDBSCAN runs the fastest in absolute terms, the *behavior* of the HDBSCAN* variants is similar, growing even slightly slower than SSDBSCAN in relative terms. In contrast, the runtime of HISSCLU’s k -cluster grows very noticeably at a much faster rate (note the log-log-scale). The different variants of HDBSCAN* are practically indistinguishable.

6.5 Final remarks

In this chapter we proposed a label-based function for FOSC framework. The results obtained in the experiments shows that our new proposal is quite competitive when compared to the methods already described in the literature, specially the constrained approach FOSC, HISSCLU and SSDBSCAN.

SSDBSCAN and HISSCLU’s k -cluster provide competitive results only for “larger” fractions of pre-labeled objects, where “large” has shown to be data-dependent and cannot be

known in advance. Besides, these algorithms perform poorly unless all class labels are guaranteed to be represented in the subset of pre-labeled objects, which is a *classification* assumption and not an adequate assumption in the semi-supervised *clustering* scenario. If this assumption is met, then SSDBSCAN may be preferred as a much faster alternative to k -cluster, but in the general case we do not recommend either algorithm for semi-supervised *clustering* scenarios where there may be unknown classes yet to be discovered and, accordingly, not represented by any pre-labeled object. In these scenarios, we recommend one of the semi-supervised HDBSCAN* variants.

When comparing our proposed label-based variant HDBSCAN*(MixBC) with its existing constraint-based counterpart HDBSCAN* (MixCON) from [Campello et al. \(2015\)](#), our label-based version has systematically provided better results, and has the advantage of working directly with labels. When comparing our label-based non-mixed variant HDBSCAN*(BC) with its constraint-based counterpart HDBSCAN*(CON) from [Campello et al. \(2015\)](#), all our results suggest that the performances of these two methods are very close to each other, but the former has the advantage of working directly with labels. In summary, irrespective of whether a mixed or a non-mixed approach is used, there is no evidence to suggest that it is worth the additional effort of producing pairwise constraints from labels, rather than working with labels directly as we propose. The mixed approach, in particular our proposed HDBSCAN*(MixBC), has shown to be more robust to unrepresented class labels, which led to a best rank performance overall when compared to all other algorithms.

APPLYING SEMI-SUPERVISED LEARNING FOR STRUCTURE-ACTIVITY ANALYSIS

When performing SAR analysis, research from the MC area faces a series of issues commonly found in data mining, such as high dimensionality and the small number of labeled objects in their data. For the first scenario, they usually apply supervised methods for feature selection, that can reduce the space of attributes, and improve the learning process, while there was not enough effort to solve the second problem. Recently, studies are using semi-supervised learning approaches aiming to use the unlabeled information, abundant in MC area, to improve the learning of classification/regression methods during the SAR analysis.

To insert the algorithms developed/used in this thesis, we performed a series of studies relative to the proper application of supervised feature selection algorithms and semi-supervised label propagation algorithms for SAR analysis. We propose three different studies where (i) we perform the propagation of labels on the unlabeled data set before using the feature selection algorithm, (ii) we perform feature selection using the labeled data provided by the user and then propagate the labels on the unlabeled data given the select features. Finally, we (iii) propose a combination between the feature selection and label propagation algorithm, where we try to acquire the set of features that provides the higher number of propagated labels.

In this chapter we describe the studied frameworks for SAR analysis. We also describe the experimental setup, the results and discussion for each study, evaluating our framework for density-based classification in terms of effectiveness.

7.1 Experimental setup

Before presenting the frameworks, we define the setup used in all experiments described in this Chapter, such as datasets, selection of partial label information, algorithms for label propagation and for classification, methods for feature selection, among others.

7.1.1 Datasets

Table 7 – List of chemical datasets collected to perform the semi-supervised classification experiments.

Dataset		#obj	#att	#biol. act
ACE	(SUTHERLAND; O'BRIEN; WEAVER, 2004)	114	16,278	[2.10, 9.90]
BZR	(SUTHERLAND; O'BRIEN; WEAVER, 2004)	163	16,288	[5.50, 8.90]
COX2	(SUTHERLAND; O'BRIEN; WEAVER, 2004)	322	16,286	[4.00, 9.00]
DHFR	(SUTHERLAND; O'BRIEN; WEAVER, 2004)	397	16,284	[3.30, 9.80]
FONTAINE	(FONTAINE <i>et al.</i> , 2005)	435	16,290	*
MIC	(GAULTON <i>et al.</i> , 2017)	219	16,271	*
PPARD121	(MALTAROLLO, 2013)	121	654	[5.95, 9.27]
TGFB	(ARAÚJO <i>et al.</i> , 2015)	59	1,719	[4.96, 7.92]

Source: Elaborated by the author.

The chemical datasets used in SAR studies are summarized in Table 7¹. We obtained the datasets “PPARD121” and “TGFB” after requesting them from the authors. The datasets had their descriptors already computed by the authors, using the online software edragon 1.0². The remaining datasets are publicly available for downloading. In these cases, we transformed the molecules to numeric attributes using the online chemical framework – Ochem³. We pre-processed the molecules of each dataset applying the default options of the software, such as standardization, neutralization, salt removal, structure cleaning, and optimization⁴. We computed E-state atom indices, Alogps, and the set of descriptors provided by the tool *PyDescriptor* (MASAND; RASTIJA, 2017), also available in the Ochem framework.

The datasets “ACE”, “BZR”, “COX2”, “DHFR”, “PPARD121”, and “TGFB” have their biological activity represented by continuous values. To categorize the biological activity to use in the semi-supervised classification scenario, we followed the same strategy applied in Rivera-Borroto *et al.* (2011), where they define the class of a molecule given the median value of the biological activity. For the molecules with biological activity higher or equal to the median they receive the active class, while molecules with biological activity lower than the median receive the inactive class. By using this approach, the class distribution of the datasets become balanced, avoiding problems during the evaluation.

After computing the chemical descriptors and categorizing the activity, we pre-processed the datasets, removing features with zero variance and with missing values higher than 10% of the dataset size. The cases where the number of missing values were lower than 10% of the

¹ We started the studies after collecting 13 datasets. However, 5 of them did not present performance values higher than 0.65 for all the experiments that we performed in this chapter. We considered values lower than 0.65 not worth discussing, so we removed from our text.

² Public software available online at <http://www.vcclab.org/>

³ Public software available available at <https://ochem.eu/>

⁴ We performed the optimization using the OpenBabel option available in Ochem software

dataset size, we imputed the missed values with the mean value of the feature. Finally, we applied the Euclidean distance to perform the experiments in the new set of attributes.

7.1.2 Data partitioning and partial information

For a better evaluation of the semi-supervised classification algorithms, we partitioned each dataset using the Stratified k-fold cross validation, using 10 folds as a parameter. In the experiments, we used 9-folds (training set) to perform the random selection of the labeled objects (partial information), propagation of the labels, and training of the classifier. We use the remaining fold (test set) to evaluate the quality of the classifier.

For the semi-supervision, we selected labeled objects from the training set, ensuring that there is at least one label from each class, repeating the random selection 20 times, thus resulting in 20 variants of each training set for each percentage of labeled objects. To study the influence of the amount of labeled data, we defined the values of 5%, 10%, and 20% of the training data⁵.

7.1.3 Feature selection

We use the propagated labels and/or the partial information obtained from the training set to perform the feature selection. We tested the ability of the Fisher Score (FS) method (DUDA; HART; STORK, 2001) to select chemical descriptors able to generate a robust classification model. The Fisher Score is a very used method in compound mining studies (ARAÚJO *et al.*, 2015; MALTAROLLO, 2013). It determines the features that characterize or separate a given data set in two or more groups. The weight given for each feature f_i considers the ratio between distances of observations in different classes, that should be as large as possible, and the distances between observations in the same class, that should be minimal as possible. Formally, given the class labels of n samples $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$ come from c classes, the Fisher Score selects the features such that the feature values of samples within the same class are small while the feature values of samples from different classes are large. The Fisher score of each feature f_i is evaluated as

$$fisherScore(f_i) = \frac{\sum_{j=1}^c n_j (\mu_{i,j} - \mu_i)^2}{\sum_{j=1}^c n_j \sigma_{i,j}^2}, \quad (7.1)$$

where n_j , μ_i , $\mu_{i,j}$ and $\sigma_{i,j}^2$ indicate the number of samples in class j , mean value of feature f_i , mean of feature f_i for the samples in class j and variance value of feature f_i for samples in class j , respectively.

Besides FS, in our experiments we also use a semi-supervised feature selection algorithm, named Locality Sensitive Discriminant Feature (LSDF), proposed by (ZHAO; LU; HE,

⁵ In Chapter 5, we defined 2%, 5%, 8%, and 10%. We attempted to verify the influence given a higher percentage of labeled objects.

2008), which uses both labeled and unlabeled data. The algorithm attempts to discover both the geometrical and discriminant structure in the data. It constructs two graphs, i.e. *within-class* and *between-class* graphs. The within-class graph connects objects that share the same label or are sufficiently close to each other, while the between-class graph connects objects which are sufficiently close to each other but have different labels (ZHAO; LU; HE, 2008). The most LSDFs are selected such that the within-class and between-class graph structures can be best preserved on these features. An attribute is assumed to be a good one if at this dimension nearby points, or points sharing the same label, are close to each other, while points with different labels are far apart (ZHAO; LU; HE, 2008). Mathematically, the within-class graph is defined as

$$S_{w_{ij}} = \begin{cases} \gamma & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \in \mathbf{X}_L \text{ and } class(\mathbf{x}_i) = class(\mathbf{x}_j), \\ 1 & \text{if } \mathbf{x}_i \text{ or } \mathbf{x}_j \in \mathbf{X}_U \text{ and } \mathbf{x}_i \in KNN(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in KNN(\mathbf{x}_i), \\ 0 & \text{otherwise.} \end{cases} \quad (7.2)$$

In Equation 7.2, the weight between two objects $\mathbf{x}_i$ and $\mathbf{x}_j$ will receive a user specified parameter γ if they belong to the set of labeled objects $\mathbf{X}_L$ and also if they have the same class label. The second condition of Equation 7.2 is that two objects $\mathbf{x}_i$ and $\mathbf{x}_j$ will receive a value of 1 if one of them belongs to the set of unlabeled objects $\mathbf{X}_U$ and $\mathbf{x}_i$ belongs to the neighborhood of $\mathbf{x}_j$ or $\mathbf{x}_j$ belongs to the neighborhood of $\mathbf{x}_i$ (the number of neighbors is also a user-specified parameter). At least, if none of the cases are satisfied, the weight between two objects will receive a weight zero. The between-class graph, is mathematically defined as

$$S_{b_{ij}} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \in \mathbf{X}_L \text{ and } class(\mathbf{x}_i) \neq class(\mathbf{x}_j), \\ 0 & \text{otherwise.} \end{cases} \quad (7.3)$$

Equation 7.3 considers given the weight 1 for the cases where two objects $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the set of labeled objects but their class labels are different. For the remaining cases, the weight between the objects will be 0. Based on Equations 7.2 and 7.3, Zhao, Lu and He (2008) compute the Laplacian graph for the within class, defined as

$$\mathbf{L}_w = \mathbf{D}_w - \mathbf{S}_w, \mathbf{D}_w = diag(\mathbf{S}_w \mathbf{1}),$$

and for the between class as

$$\mathbf{L}_b = \mathbf{D}_b - \mathbf{S}_b, \mathbf{D}_b = diag(\mathbf{S}_b \mathbf{1}),$$

where

$$\mathbf{1} = (1, 1, \dots, 1)^T.$$

Given the Laplacian graphs for the within and between classes, it is possible to define the weight of each attribute using the ration between $\mathbf{L}_b$ and $\mathbf{L}_w$, which will be defined as

$$L_r = \frac{\mathbf{f}_r^T \mathbf{L}_b \mathbf{f}_r}{\mathbf{f}_r^T \mathbf{L}_w \mathbf{f}_r},$$

for $r = 1, 2, \dots, d$. During experiments, we ordered the features scores according to their importance and selected different number of features, in the interval $[5, 30)$. In general, the algorithms presented good results for the number of features equals to 25 for both the Fisher score and LSDF. For this reason, all results presented in this document are results obtained after selecting 25 variables using FS and LSDF.

The LSDF algorithm requires two extra parameters such as the weight between labeled objects with the same class γ , and the neighborhood parameter. For these cases, we adopted the values already used in Zhao, Lu and He (2008), with $\gamma = 100$ and $k = 5$.

7.1.4 Algorithms and parameters

In this experiments, we used the algorithms HDBSCAN*(cd,-), HDBSCAN*(ap,-), HDBSCAN*(cd,wMST) and RMGT. For the last two algorithms we apply exactly the same parameters used in the experiments described in section 5.4, since the algorithms presented the best results in the experiments described in section 5.5. For the algorithm HDBSCAN*(cd,-) we define $m_{\text{pts}} = 4$.⁶

We combined the partial information, the unlabeled data and the set of propagated labels of each algorithm to train a Support Vector Classifier (SVC) (BOSER; GUYON; VAPNIK, 1992; CORTES; VAPNIK, 1995) with default parameters, such as Radial basis function as kernel and penalty parameter ($C = 1.0$). For the sake of comparison, we also modeled the SVC with the partial information. With this scenario, the actual influence of the semi-supervised classification algorithms can be observed on the structure-activity analysis.

7.1.5 Validation

Since most of the datasets are balanced and binary (we do not have a problem with multiclass datasets in these experiments), we report the accuracy computed on the labels predicted by the SVC algorithm. In this case, we do not have influence of those objects whose labels have not been exposed to the semi-supervised classification algorithms for training. Since the number of datasets is relatively small, we do not apply the statistical validation described in subsection 5.4.5.

7.1.6 Source code

We carried out the experiments using two programming languages. The programming language Java was used for most of the semi-supervised classification algorithms. The programming language Python was used to control processes such as data pre-processing, random selection of the partial information, data partitioning, classifier training and model evaluation. All the codes are made available after requesting them from the author.

⁶ We also use in this Chapter the same parameter used in section 5.4.

7.2 Propagating labels for feature selection

We describe in this section the first scheme implemented for SAR analysis. In Figure 31, we present one possible combination of semi-supervised classification algorithms and feature selection. Given a dataset with labeled and unlabeled data, we perform the propagation of the labels to the unlabeled data using a semi-supervised classification algorithm. Then, using the new set of labeled data, we perform the use of a feature selection algorithm. At least, using the propagated labels and the selected attributes, we train a classifier which will become able to predict new compounds. Our assumption in this scenario is that, given a higher amount of labeled data for the feature selection algorithm, we can acquire the best set of attributes that will improve the efficacy of a fully supervised classification algorithm.

Figure 31 – Diagram for the first framework specified for SAR analysis. The steps involve using of a label propagation algorithm before the application of the variable selection method.

Source: Elaborated by the author.

Figure 32 – Summary for the experiments presented in the section, which refers to the label propagation and feature selection using the propagated labels.

Source: Elaborated by the author.

Figure 32 presents how the algorithms were fitted into the framework of Figure 31. We defined as the baseline both FS and LSDF to compare them with the other variations of the framework. For both FS and LS, we use the partial information, represented by the set of labeled objects obtained from the training set, for feature selection. In the results, we compare FS (baseline) with HDBSCAN(cd, wMST)-FS and RMGT-FS, while LSDF (baseline) we compare with HDBSCAN(cd, wMST)-LSDF and RMGT-LSDF. This configuration provides a real perspective regarding the effect of the semi-supervised classification algorithms.

7.2.1 Results

Table 8 – Average results (accuracy) obtained for each dataset presented in Table 7, using the label propagation and feature selection. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors **green**, to represent an improvement, and **red** to represent a drop.

#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
ACE	FS	0.52/0.00	0.67/0.00	0.81/0.00
	HDBSCAN*(cd,wMST)-FS	0.55/ 5.77	0.69/ 2.99	0.81/0.00
	RMGT-FS	0.54/ 3.85	0.70/ 4.48	0.82/ 1.23
	LSDF	0.56/0.00	0.68/0.00	0.80/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.55/ -1.79	0.68/0.00	0.79/ -1.25
	RMGT-LSDF	0.55/ -1.79	0.70/ 2.94	0.81/ 1.25
BZR	FS	0.51/0.00	0.55/0.00	0.61/0.00
	HDBSCAN*(cd,wMST)-FS	0.51/0.00	0.53/ -3.64	0.58/ -4.92
	RMGT-FS	0.51/0.00	0.54/ -1.82	0.61/0.00
	LSDF	0.51/0.00	0.55/0.00	0.60/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.51/0.00	0.53/ -3.64	0.57/ -5.00
	RMGT-LSDF	0.51/0.00	0.54/ -1.82	0.60/0.00
COX2	FS	0.56/0.00	0.62/0.00	0.66/0.00
	HDBSCAN*(cd,wMST)-FS	0.55/ -1.79	0.60/ -3.23	0.65/ -1.52
	RMGT-FS	0.55/ -1.79	0.60/ -3.23	0.66/0.00
	LSDF	0.56/0.00	0.61/0.00	0.66/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.54/ -3.57	0.59/ -3.28	0.65/ -1.52
	RMGT-LSDF	0.55/ -1.79	0.60/ -1.64	0.65/ -1.52

Continued on next page...

continued from previous page				
#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
DHFR	FS	0.54/0.00	0.59/0.00	0.66/0.00
	HDBSCAN*(cd,wMST)-FS	0.54/0.00	0.56/-5.08	0.63/-4.55
	RMGT-FS	0.53/-1.85	0.57/-3.39	0.64/-3.03
	LSDF	0.54/0.00	0.59/0.00	0.65/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.53/-1.85	0.56/-5.08	0.63/-3.08
	RMGT-LSDF	0.53/-1.85	0.57/-3.39	0.63/-3.08
FONTAINE	FS	0.66/0.00	0.70/0.00	0.76/0.00
	HDBSCAN*(cd,wMST)-FS	0.64/-3.03	0.70/0.00	0.75/-1.32
	RMGT-FS	0.65/-1.52	0.70/0.00	0.74/-2.63
	LSDF	0.66/0.00	0.71/0.00	0.77/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.64/-3.03	0.69/-2.82	0.76/-1.30
	RMGT-LSDF	0.64/-3.03	0.70/-1.41	0.76/-1.30
MIC	FS	0.60/0.00	0.71/0.00	0.77/0.00
	HDBSCAN*(cd,wMST)-FS	0.59/-1.67	0.71/0.00	0.77/0.00
	RMGT-FS	0.58/-3.33	0.71/0.00	0.77/0.00
	LSDF	0.60/0.00	0.71/0.00	0.77/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.58/-3.33	0.70/-1.41	0.76/-1.30
	RMGT-LSDF	0.58/-3.33	0.71/0.00	0.76/-1.30
PPARD121	FS	0.58/0.00	0.62/0.00	0.66/0.00
	HDBSCAN*(cd,wMST)-FS	0.58/0.00	0.62/0.00	0.65/-1.52
	RMGT-FS	0.58/0.00	0.62/0.00	0.66/0.00
	LSDF	0.58/0.00	0.62/0.00	0.65/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.58/0.00	0.62/0.00	0.65/0.00
	RMGT-LSDF	0.58/0.00	0.63/1.61	0.65/0.00
TGFB	FS	0.55/0.00	0.59/0.00	0.64/0.00
	HDBSCAN*(cd,wMST)-FS	0.55/0.00	0.59/0.00	0.64/0.00
	RMGT-FS	0.55/0.00	0.59/0.00	0.64/0.00
	LSDF	0.55/0.00	0.59/0.00	0.64/0.00
	HDBSCAN*(cd,wMST)-LSDF	0.55/0.00	0.59/0.00	0.63/-1.56
	RMGT-LSDF	0.55/0.00	0.59/0.00	0.64/0.00

Source: Research data.

Table 8 shows the results obtained from the datasets described in Table 7. We present the accuracy and the percentage of variation obtained with respect to the baseline algorithm. We observed in this scenario that none of the algorithms for label propagation benefit the feature selection algorithm/classification model during their learning process. In fact, the results given in Table 8 suggest that we should not use the label propagation algorithms before the attribute selection, since we can degenerate the learning process.

The dataset “ACE” is the only one that presents a small improvement using the composition of label propagation and feature selection methods, for all the percentages of labeled objects. The best result for 5% of labeled objects was obtained using only the LSDF algorithm while the best results for 10% and 20% was obtained by the RMGT-FS with an improvement of 4.48% and 1.23% compared to the baseline (FS), respectively. The remaining datasets show a negative or no variation with respect to their baselines.

One simple explanation about the results in Table 8 is that the HDBSCAN*(cd, wMST) and RMGT also suffer from the high dimensionality problem, since they have to compute distances between objects to produce their respective graphs. Once the distances between objects become close as the dimension increases, it is difficult to define areas that can separate objects from different classes.

Considering that the results are not enough to assist the SAR analysis, we carried out other experiments using a different combination between label propagation and feature selection algorithms, which will be described in the next section.

7.3 Selecting features for label propagation

We describe in this section the second scheme implemented for SAR analysis. In Figure 33 we present another possible combination of semi-supervised classification algorithms and feature selection methods. Given a dataset (labeled and unlabeled data), we execute the feature selection algorithms using the partial information. Then, we perform the label propagation using the data with the subset of features previously selected. At least, we follow the same statement described in the previous section, by using the propagated labels and the selected attributes to train a classifier which will be able to predict new compounds. Our assumption in this scenario is that, a higher amount of labeled data can help, in fact, the fully supervised classification technique instead of the feature selection methods.

Figure 34 presents how the algorithms were fitted in the framework shown in Figure 33. Here, again, we defined as the baseline both FS and LSDF to compare with the other variations of the framework. As in the previous experiment, for both FS and LS we use the partial information for the feature selection. In the results, we compare FS (baseline) with FS-HDBSCAN(cd, wMST) and FS-RMGT, while LSDF (baseline) we compare with LSDF-HDBSCAN(cd, wMST) and LSDF-RMGT.

Figure 33 – Diagram for the second framework defined for SAR analysis. The steps involve using of a feature selection method before the application of the label propagation algorithm.

Source: Elaborated by the author.

Figure 34 – Summary for the experiments presented in the section, which refers to the feature selection and label propagation procedures.

Source: Elaborated by the author.

7.3.1 Results

Table 9 shows the results obtained from the datasets described in Table 7. We present the accuracy and the percentage of variation obtained w.r.t., the baseline algorithm. Overall, for all the datasets, an improvement of the results can be observed, suggesting one possible configuration of semi-supervised classification algorithms for SAR analysis.

Table 9 – Average results (accuracy) obtained for each dataset presented in Table 7, considering the framework presented in Figure 34. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors green, to represent an improvement, and red to represent a drop in performance.

#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
ACE	FS	0.52/0.00	0.67/0.00	0.81/0.00
	FS-HDBSCAN*(cd,wMST)	0.68/30.77	0.74/10.45	0.82/1.23
	FS-RMGT	0.67/28.85	0.74/10.45	0.81/0.00
	LSDF	0.56/0.00	0.68/0.00	0.80/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.72/28.57	0.74/8.82	0.81/1.25
	LSDF-RMGT	0.70/25.00	0.74/8.82	0.81/1.25
BZR	FS	0.51/0.00	0.55/0.00	0.61/0.00
	FS-HDBSCAN*(cd,wMST)	0.56/9.80	0.60/9.09	0.64/4.92
	FS-RMGT	0.56/9.80	0.62/12.73	0.65/6.56
	LSDF	0.51/0.00	0.55/0.00	0.60/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.56/9.80	0.58/5.45	0.63/5.00
	LSDF-RMGT	0.56/9.80	0.60/9.09	0.64/6.67
COX2	FS	0.56/0.00	0.62/0.00	0.66/0.00
	FS-HDBSCAN*(cd,wMST)	0.61/8.93	0.63/1.61	0.66/0.00
	FS-RMGT	0.62/10.71	0.64/3.23	0.66/0.00
	LSDF	0.56/0.00	0.61/0.00	0.66/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.59/5.36	0.62/1.64	0.66/0.00
	LSDF-RMGT	0.61/8.93	0.64/4.92	0.66/0.00
DHFR	FS	0.54/0.00	0.59/0.00	0.66/0.00
	FS-HDBSCAN*(cd,wMST)	0.61/12.96	0.66/11.86	0.70/6.06
	FS-RMGT	0.61/12.96	0.67/13.56	0.70/6.06
	LSDF	0.54/0.00	0.59/0.00	0.65/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.61/12.96	0.65/10.17	0.69/6.15
	LSDF-RMGT	0.61/12.96	0.66/11.86	0.69/6.15

Continued on next page...

...continued from previous page

#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
FONTAINE	FS	0.66/0.00	0.70/0.00	0.76/0.00
	FS-HDBSCAN*(cd,wMST)	0.74/12.12	0.78/11.43	0.80/5.26
	FS-RMGT	0.75/13.64	0.78/11.43	0.80/5.26
	LSDF	0.66/0.00	0.71/0.00	0.77/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.75/13.64	0.79/11.27	0.82/6.49
	LSDF-RMGT	0.75/13.64	0.80/12.68	0.82/6.49
MIC	FS	0.60/0.00	0.71/0.00	0.77/0.00
	FS-HDBSCAN*(cd,wMST)	0.73/21.67	0.75/5.63	0.78/1.30
	FS-RMGT	0.72/20.00	0.76/7.04	0.78/1.30
	LSDF	0.60/0.00	0.71/0.00	0.77/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.72/20.00	0.76/7.04	0.79/2.60
	LSDF-RMGT	0.72/20.00	0.76/7.04	0.78/1.30
PPARD121	FS	0.58/0.00	0.62/0.00	0.66/0.00
	FS-HDBSCAN*(cd,wMST)	0.59/1.72	0.63/1.61	0.65/-1.52
	FS-RMGT	0.60/3.45	0.64/3.23	0.65/-1.52
	LSDF	0.58/0.00	0.62/0.00	0.65/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.60/3.45	0.62/0.00	0.66/1.54
	LSDF-RMGT	0.61/5.17	0.64/3.23	0.66/1.54
TGFB	FS	0.55/0.00	0.59/0.00	0.64/0.00
	FS-HDBSCAN*(cd,wMST)	0.55/0.00	0.61/3.39	0.67/4.69
	FS-RMGT	0.58/5.45	0.62/5.08	0.70/9.38
	LSDF	0.55/0.00	0.59/0.00	0.64/0.00
	LSDF-HDBSCAN*(cd,wMST)	0.59/7.27	0.62/5.08	0.68/6.25
	LSDF-RMGT	0.59/7.27	0.63/6.78	0.69/7.81

Source: Research data.

The first observation given the results, is that the dataset ACE showed the best improvement rates when using a label propagation algorithm after feature selection. Besides ACE, the MIC and FONTAINE datasets also present great improvement rates when compared to their respective baselines. For the label propagation algorithms, we verified that in general, without too much difference from HDBSCAN*(cd, wMST), the combination between any feature selection

method and RMGT produced the best results in our experiments, specially for 10% of the labeled data. One possible explanation about this could be the absence of density-based cluster structure for the label propagation using HDBSCAN*(cd, wMST). With respect to the baseline methods (FS and LSDF), we observed that they obtained similar performance. The LSDF algorithm did not use the unlabeled data efficiently to improve the feature selection results.

In this new study, we obtained a considerable improvement in the results, when comparing to study performed in the previous section (Table 8). Although there was an improvement, we decided to perform a completely different combination of algorithms for the SAR analysis. The new study adopts the feature selection methods with SSDBSCAN++. The next section describes the new framework and the results in more detail.

7.4 Combining Feature selection and SSDBSCAN++

In this section, we describe a totally different approach for SAR analysis. We attempted to combine a feature selection algorithm with the semi-supervised clustering algorithm SSDBSCAN++, described in subsection 6.4.1. SSDBSCAN++ algorithm tends to identify noise objects during their label expansion. Such noise objects are located in areas where the density level allows them to connect to labeled objects with different class labels.

Our novel approach is presented in Figure 35. It evolves, as in the previous section, the feature selection using the set of labeled objects existing in the training set. After the feature selection phase, we perform the label expansion using SSDBSCAN++. Since SSDBSCAN++ identifies noise objects during its label expansion, we can have cases where the unlabeled data are not entirely labeled, which suggests that the space produced by the set of selected features would not be enough to define a low density area able to separate objects with different class labels. For this reason, we return to the feature selection step to perform the choice of a new set of attributes using, at this moment, the result obtained by SSDBSCAN++. We can perform this step iteratively, until all the unlabeled data become labeled, or reach a maximum number of attempts. In the latter case, we store the maximum number of objects labeled by SSDBSCAN++. With respect to the features, we store the ones correspondent to the higher number of labeled objects.

In our experiments, we execute SSDBSCAN++ with two different configurations: (i) computing the core distance (Definition 5, named here as SSDBSCAN++(cd), and (ii) using the definition of all-points-core-distance (Definition 10), named here as SSDBSCAN++(ap). For SSDBSCAN++(cd) we defined $m_{pts} = 4$, the same parameter used for the semi-supervised classification experiments.

Figure 36 presents how the algorithms were adjusted in the framework shown in Figure 35. Here, we also defined as the baseline both FS and LSDF for comparison with the other variations of the framework. As in the previous experiment, for both FS and LS we use

Figure 35 – Diagram for the third framework defined for SAR analysis. The framework combines both semi-supervised clustering and feature selection techniques for label propagation and selection of relevant features.

Source: Elaborated by the author.

Figure 36 – Summary for the experiments presented in the section, which refers to the feature selection and label propagation procedures.

Source: Elaborated by the author.

the partial information for feature selection. In the results, we compare FS (baseline) with FS-SSDBSCAN++(ap) and FS-SSDBSCAN++(cd), while LSDF (baseline) we compare with LSDF-SSDBSCAN++(ap) and LSDF-SSDBSCAN++(cd).

7.4.1 Results

Table 10 shows the results obtained from the datasets described in Table 7. We present the accuracy and the percentage of variation obtained w.r.t., the baseline algorithm. Overall, for all the datasets, an improvement of the results can be observed, as seen in the study described in section 7.3, suggesting a new configuration of semi-supervised clustering algorithms for SAR

analysis.

Table 10 – Average results (accuracy) obtained for each dataset presented in Table 7, using the feature selection combined with the SSDBSCAN++ algorithm. We show the results for each percentage of labeled objects and the percentage of variation with respect to the baseline, with the colors **green**, to represent an improvement, and **red** to represent a drop in performance.

#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
ACE	FS	0.52/0.00	0.69/0.00	0.81/0.00
	FS-SSDBSCAN++(ap)	0.66/ 26.92	0.76/ 10.14	0.81/0.00
	FS-SSDBSCAN++(cd)	0.66/ 26.92	0.75/ 8.70	0.81/0.00
	LSDF	0.52/0.00	0.69/0.00	0.81/0.00
	LSDF-SSDBSCAN++(ap)	0.72/ 38.46	0.79/ 14.49	0.82/ 1.23
	LSDF-SSDBSCAN++(cd)	0.72/ 38.46	0.79/ 14.49	0.83/ 2.47
BZR	FS	0.49/0.00	0.55/0.00	0.60/0.00
	FS-SSDBSCAN++(ap)	0.55/ 12.24	0.60/ 9.09	0.61/ 1.67
	FS-SSDBSCAN++(cd)	0.54/ 10.20	0.58/ 5.45	0.61/ 1.67
	LSDF	0.49/0.00	0.55/0.00	0.60/0.00
	LSDF-SSDBSCAN++(ap)	0.58/ 18.37	0.58/ 5.45	0.62/ 3.33
	LSDF-SSDBSCAN++(cd)	0.55/ 12.24	0.58/ 5.45	0.61/ 1.67
COX2	FS	0.54/0.00	0.64/0.00	0.66/0.00
	FS-SSDBSCAN++(ap)	0.60/ 11.11	0.64/0.00	0.64/ -3.03
	FS-SSDBSCAN++(cd)	0.60/ 11.11	0.65/ 1.56	0.63/ -4.55
	LSDF	0.54/0.00	0.64/0.00	0.66/0.00
	LSDF-SSDBSCAN++(ap)	0.60/ 11.11	0.62/ -3.12	0.65/ -1.52
	LSDF-SSDBSCAN++(cd)	0.59/ 9.26	0.64/0.00	0.64/ -3.03
DHFR	FS	0.56/0.00	0.58/0.00	0.64/0.00
	FS-SSDBSCAN++(ap)	0.59/ 5.36	0.65/ 12.07	0.67/ 4.69
	FS-SSDBSCAN++(cd)	0.62/ 10.71	0.67/ 15.52	0.68/ 6.25
	LSDF	0.56/0.00	0.58/0.00	0.64/0.00
	LSDF-SSDBSCAN++(ap)	0.64/ 14.29	0.66/ 13.79	0.70/ 9.38
	LSDF-SSDBSCAN++(cd)	0.63/ 12.50	0.66/ 13.79	0.68/ 6.25

Continued on next page. . .

... continued from previous page

#Dataset	#Algorithm	#Perc labeled / #Imp w.r.t. baseline		
		5%	10%	20%
FONTAINE	FS	0.68/0.00	0.73/0.00	0.72/0.00
	FS-SSDBSCAN++(ap)	0.72/5.88	0.74/1.37	0.76/5.56
	FS-SSDBSCAN++(cd)	0.71/4.41	0.75/2.74	0.77/6.94
	LSDF	0.68/0.00	0.73/0.00	0.72/0.00
	LSDF-SSDBSCAN++(ap)	0.73/7.35	0.77/5.48	0.80/11.11
	LSDF-SSDBSCAN++(cd)	0.72/5.88	0.76/4.11	0.79/9.72
MIC	FS	0.60/0.00	0.70/0.00	0.77/0.00
	FS-SSDBSCAN++(ap)	0.70/16.67	0.76/8.57	0.78/1.30
	FS-SSDBSCAN++(cd)	0.71/18.33	0.76/8.57	0.78/1.30
	LSDF	0.60/0.00	0.70/0.00	0.77/0.00
	LSDF-SSDBSCAN++(ap)	0.72/20.00	0.77/10.00	0.79/2.60
	LSDF-SSDBSCAN++(cd)	0.70/16.67	0.77/10.00	0.78/1.30
PPARD121	FS	0.59/0.00	0.63/0.00	0.64/0.00
	FS-SSDBSCAN++(ap)	0.60/1.69	0.66/4.76	0.64/0.00
	FS-SSDBSCAN++(cd)	0.61/3.39	0.64/1.59	0.66/3.12
	LSDF	0.59/0.00	0.63/0.00	0.64/0.00
	LSDF-SSDBSCAN++(ap)	0.61/3.39	0.64/1.59	0.65/1.56
	LSDF-SSDBSCAN++(cd)	0.62/5.08	0.65/3.17	0.67/4.69
TGFB	FS	0.55/0.00	0.58/0.00	0.65/0.00
	FS-SSDBSCAN++(ap)	0.60/9.09	0.63/8.62	0.66/1.54
	FS-SSDBSCAN++(cd)	0.58/5.45	0.62/6.90	0.66/1.54
	LSDF	0.55/0.00	0.58/0.00	0.65/0.00
	LSDF-SSDBSCAN++(ap)	0.61/10.91	0.63/8.62	0.69/6.15
	LSDF-SSDBSCAN++(cd)	0.61/10.91	0.64/10.34	0.68/4.62

Source: Research data.

Given the results in Table 10, we verify that the dataset ACE, again, presented the best improvement rates using the new framework. Besides ACE, the datasets BZR, COX2 DHFR, MIC and TGFB datasets also presents great improvement rates when comparing to their respective

baselines, specifically for the cases where the percentage of labeled objects for training is small (5% of labeled objects). We also observed that in general, LSDF-SSDBSCAN++(ap) performed the best results, although the results are not so different from the remaining algorithms. An explanation about this phenomena could be the parameterless property of SSDBSCAN++(ap).

7.5 Final remarks

In this chapter, we provide a series of frameworks for SAR analysis. With the experimental setup, we observed that performing feature selection and then the label propagation with a semi-supervised classification algorithm provided an improvement during the SAR analysis, specially in scenarios where the number of labeled objects is insufficient.

Another study that also improves the SAR analysis is related to the use of a semi-supervised clustering and feature selection algorithm to define both the set of attributes and the maximum number of features for training of a fully supervised classifier. In this scenario, we used the semi-supervised clustering algorithm SSDBSCAN++ to perform the label expansion step. Both studies presented in this chapter can be extended for tests with other datasets from the MC area.

CONCLUSIONS AND FUTURE WORK

In this thesis, we explored different aspects regarding semi-supervised learning algorithms, which includes (i) developing of a new framework for semi-supervised classification using density-based clustering algorithms, (ii) adding a label-based optimal cluster extraction from hierarchies, and (iii) a study of semi-supervised learning algorithms applied to the analysis of relationships between chemical structure and biological activity. Below, we provide our concluding remarks regarding each one of these topics.

In the context of semi-supervised classification, we presented a unified view of density-based clustering algorithms that gives rise to a framework of semi-supervised density-based classification. This framework was obtained after extending the state-of-the-art hierarchical clustering algorithm HDBSCAN*, described initially as a unsupervised and constraint-based semi-supervised clustering algorithm. We show that HISSCLU and SSDBSCAN, important algorithms from semi-supervised clustering literature, can also be derived as particular cases from our framework, with the advantage that it eliminates possible shortcomings of these traditional algorithms, making it simpler and easier to interpret. The building blocks contained in our framework produces a huge number of novel variants of algorithms which have never been tried before. At least, we showed experimentally that several instances of the proposed framework can achieve comparable or better quality than HISSCLU and other traditional methods from the semi-supervised classification literature, namely RMGT, LapSVM, and GFHF, while being computationally more efficient as well as interpretable from a density-based, non-parametric viewpoint.

Regarding semi-supervised clustering, we extended HDBSCAN*, which plays a central role in our unified view and framework for density-based classification, to also perform semi-supervised clustering from a collection of pre-labeled data objects, rather than pairwise constraints, already supported by the algorithm. The direct use of labels has been shown to be both simpler and more effective for semi-supervised cluster extraction from hierarchies. The results obtained in semi-supervised clustering scenarios where one or more categories in the data

are not represented in the collection of pre-labeled objects (controlled random scenario) are far superior than competitors from the literature, such as HISSCLU and SSDBSCAN. This improvement with respect to HISSCLU and SSDBSCAN is due to the fact that we can combine both supervised and unsupervised measures (mixed approach) for cluster extraction in HDBSCAN*, while HISSCLU proceeds a horizontal cut in the colored version of a reachability plot (produced during the semi-supervised classification step), which has already been shown to be inefficient. In SSDBSCAN, the absence of labels makes that one cluster to be completely lost or receive a label for another cluster if there is no label information for such cluster.

Regarding the Medicinal Chemistry area, we presented a study that consists of the structure activity relationship analysis using semi-supervised learning algorithms. One of the main difficulties that we observed during the experiments was the proper way to adjust the semi-supervised learning algorithm in our framework. For this reason, we tested three different configurations for the framework. The first and second approaches consist of using label propagation algorithms before and after the feature selection procedure. The results show to be a better improvement on the results if we perform the feature selection, even with a small number of labeled objects, so that they can be applied later to a label propagation algorithm. By using this approach, we were able to achieve good improvement for the classification task.

In the last scenario, we defined a novel approach that, iteratively, combines a feature selection algorithm with a semi-supervised clustering algorithm, called SSDBSCAN++, to perform both feature selection and label propagation. In this case, we initialize the procedure by using the set of labeled objects provided by the user to perform feature selection. Then, we use the SSDBSCAN++(ap) or SSDBSCAN++(cd) to perform the label propagation procedure, given the attributes selected by the feature selection algorithm. At the end, we return to the feature selection procedure using the set of label objects obtained during the label propagation step. The results obtained with this new scenario suggest using this new framework for SAR analysis.

8.1 Future work

In semi-supervised transductive classification, a large amount of unlabeled data can be classified based on a small fraction of labeled data, which is not representative enough for training a fully supervised classification model. Many semi-supervised learning algorithms from actual literature generally require prior construction of a graph (*graph assumption*) so that the label transduction could be possible. Given the best of our efforts, we observed that there is no current work which proposes using a graph built under a density-based clustering perspective in a semi-supervised scenario for use in semi-supervised graph-based algorithms. For this reason, as future work, a possible line of research in semi-supervised classification is to use the MST_r graph (as opposed to traditional graphs such as mutual k nearest neighbors) in conjunction with other graph-based label propagation algorithms from the literature. By using/adapting this

graph, it is possible that the results obtained could be relevant and provide a new tool to help analyze the multivariate data, as well as to contribute with the state-of-the-art of semi-supervised classification.

Regarding semi-supervised clustering, we proposed a label-based approach for optimal cluster extraction from hierarchies, that showed to be quite efficient when compared to the constrained-based approach and to the traditional methods for semi-supervised clustering, such as HISSCLU and SSDBSCAN. With respect to the traditional ones, our approach presented better results, specially when there is no class information for all the clusters in the dataset, which was expected for a semi-supervised clustering scenario. The best results we obtained are due the fact that we could combine the label-based with the unsupervised method (the stability method, already presented in HDBSCAN*) for cluster extraction, called HDBSCAN*(Mix-BC).

Recently, a modularity-based measure for cluster selection from hierarchies was proposed, that showed to be as effective as the stability approach for optimal cluster extraction (ANJOS, 2018). It would be interesting to investigate if this new measure could be utilized combined with the label-based approach for cluster extraction. Another interesting topic for investigation is the problem of model selection in the semi-supervised scenario, namely, the direct use of labels rather than constraints to guide the choice of the m_{pts} parameter in HDBSCAN*. By using the semi-supervised model selection for the choice of m_{pts} , we could use this approach for all similarity-based measures, already supported by HDBSCAN*.

Regarding the Medicinal Chemistry area, we presented a series of studies with respect to the applications of semi-supervised learning algorithms for the study of relationships between chemical structure and biological activity (SAR studies). We presented results with the three frameworks for feature selection and semi-supervised learning algorithms. One of the frameworks, where we combine semi-supervised clustering algorithm (SSDBSCAN++) and the feature selection algorithms could be improved, by defining a better way to perform its iterative procedure by, for example, defining a function that identifies the convergence of the framework.

The experiments described in Chapter 7 were performed in controlled scenarios, where the unlabeled information was obtained as part of the dataset under study. For future studies, we will verify the accuracy improvement by extracting unlabeled data from public datasets of chemical compounds (i.e. ChEMBL or ZINC). Other future work consists of adapting our semi-supervised frameworks to the quantitative SAR analysis (QSAR) scenario, that consists of using regression methods to predict the biological activity of a molecule. Instead of propagating the class labels, we can direct efforts to propagate continuous values over the graph produced by the HDBSCAN* framework. During the experiments with semi-supervised classification, we identified the publication of Hasanloei *et al.* (2018), where they proposed a semi-supervised feature selection for QSAR analysis. We can use this algorithm as a baseline for our studies.

8.2 Publications and financial support

During the PhD, I have published journal and conference papers. Some of these are related to this thesis, whereas others are partially related, as presented below.

Journal papers

- **GERTRUDES, JADSON CASTRO; ZIMEK, ARTHUR; SANDER, JÖRG; CAMPELLO, RICARDO J. G. B. A Unified View of Density-based Methods for Semi-Supervised Clustering and Classification.** Data Mining and Knowledge Discovery. (**Under review**).
- **ARAÚJO, S. C.; MALTAROLLO, V.G.; SILVA, D. C; GERTRUDES, J.C.; HONÓRIO, K. M. ALK-5 Inhibition: A Molecular Interpretation of the Main Physicochemical Properties Related to Bioactive Ligands.** JOURNAL OF THE BRAZILIAN CHEMICAL SOCIETY. Impact Factor (2016 JCR): 1,1980, v.26, p.1936–1946, 2015.
- **MALTAROLLO, VINÍCIUS GONÇALVES; GERTRUDES, JADSON CASTRO; OLIVEIRA, PATRÍCIA RUFINO; HONÓRIO, KATHIA MARIA. Applying machine learning techniques for ADME-Tox prediction: a review.** Expert Opinion on Drug Metabolism & Toxicology. Impact Factor (2016 JCR): 3,0270, v.11, p.259–271, 2015.
- **PANTALEÃO, SIMONE QUEIROZ; MALTAROLLO, VINICIUS GONÇALVES; ARAÚJO, SHEILA CRUZ; GERTRUDES, JADSON CASTRO; HONÓRIO, KATHIA MARIA Molecular docking studies and 2D analyses of DPP-4 inhibitors as candidates in the treatment of diabetes.** Molecular BioSystems. Impact factor(2016 JCR): 2,7810, v.11, p.3188–3193, 2015.

Conference papers

- **GERTRUDES, JADSON CASTRO; ZIMEK, ARTHUR; SANDER, JÖRG; CAMPELLO, RICARDO J. G. B. A unified framework of density-based clustering for semi-supervised classification** In: the 30th International Conference, 2018, Bozen-Bolzano. Proceedings of the 30th International Conference on Scientific and Statistical Database Management – SSDBM '18. New York: ACM Press, 2018. p.1–12
- **ANJOS, F. A. R.; GERTRUDES, JADSON CASTRO; CAMPELLO, RICARDO J. G. B.; SANDER, JÖRG. A Modularity-Based Measure for Cluster Selection from Clustering Hierarchies.** In: Australasian Data Mining Conference (AusDM), 2018, Bathurst. Proceedings of the 16th Australasian Data Mining Conference. Springer, 2018.

Financial support

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Finance Code 001. During the Ph.D. sandwich conducted at University of Alberta (Canada) for 11 months (01/10/2015 until 31/08/2016) I also received support from the National Council for Scientific and Technological Development (CNPq), process number 202883/2015-9.

BIBLIOGRAPHY

ACKERMAN, M.; BEN-DAVID, S.; LOKER, D. Characterization of linkage-based clustering. In: **The 23rd Conference on Learning Theory (COLT), Haifa, Israel**. [S.l.: s.n.], 2010. p. 270–281. Citation on page [44](#).

AGGARWAL, C. C.; REDDY, C. K. **Data Clustering: Algorithms and Applications**. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2013. ISBN 1466558210, 9781466558212. Citation on page [29](#).

AMIGÓ, E.; GONZALO, J.; ARTILES, J.; VERDEJO, F. A comparison of extrinsic clustering evaluation metrics based on formal constraints. **Inf. Retr.**, v. 12, n. 4, p. 461–486, 2009. Citation on page [81](#).

ANDONIE, R. Extreme data mining: Inference from small datasets. **International Journal of Computers, Communications and Control**, v. 5, p. 280–291, 2010. Citation on page [63](#).

ANJOS, F. d. A. R. d. **Seleção de grupos a partir de hierarquias: uma modelagem baseada em grafos**. Master's Thesis (Master's Thesis) — Universidade de São Paulo, 2018. Citation on page [123](#).

ANKERST, M.; BREUNIG, M. M.; KRIEGL, H.-P.; SANDER, J. OPTICS: ordering points to identify the clustering structure. In: **Proc. ACM SIGMOD**. [S.l.: s.n.], 1999. p. 49–60. Citations on pages [35](#), [36](#), [37](#), and [52](#).

ARAÚJO, S. C.; MALTAROLLO, V. G.; SILVA, D. C.; GERTRUDES, J. C.; HONÓRIO, K. M. ALK-5 Inhibition: A Molecular Interpretation of the Main Physicochemical Properties Related to Bioactive Ligands. **Journal of the Brazilian Chemical Society**, scielo, v. 26, p. 1936 – 1946, 09 2015. ISSN 0103-5053. Citations on pages [64](#), [104](#), and [105](#).

BAGGA, A.; BALDWIN, B. Entity-based cross-document coreferencing using the vector space model. In: **36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL**. [S.l.: s.n.], 1998. p. 79–85. Citation on page [81](#).

BASU, S.; DAVIDSON, I.; WAGSTAFF, K. (Ed.). **Constrained Clustering: Advances in Algorithms, Applications and Theory**. [S.l.]: CRC Press, 2008. Citation on page [86](#).

BATISTA, A. J. L.; CAMPELLO, R. J. G. B.; SANDER, J. Active semi-supervised classification based on multiple clustering hierarchies. In: **Proc. DSAA**. [S.l.: s.n.], 2016. p. 11–20. Citation on page [23](#).

BELKIN, M.; NIYOGI, P.; SINDHWANI, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. **JMLR**, v. 7, p. 2399–2434, 2006. Citations on pages [51](#) and [74](#).

BISHOP, C. M. **Pattern Recognition and Machine Learning (Information Science and Statistics)**. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN 0387310738. Citation on page [29](#).

BÖHM, C.; PLANT, C. Hissclu: a hierarchical density-based method for semi-supervised clustering. In: **Proc. EDBT**. [S.l.: s.n.], 2008. p. 440–451. Citations on pages 52, 57, 59, 72, 75, 90, and 91.

BOSER, B. E.; GUYON, I.; VAPNIK, V. A training algorithm for optimal margin classifiers. In: **Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT 1992, Pittsburgh, PA, USA, July 27–29, 1992**. [s.n.], 1992. p. 144–152. Available: <<http://doi.acm.org/10.1145/130385.130401>>. Citation on page 107.

CAMPELLO, R. J. G. B.; MOULAVI, D.; SANDER, J. Density-based clustering based on hierarchical density estimates. In: **Proc. PAKDD**. [S.l.: s.n.], 2013. p. 160–172. Citations on pages 36, 38, 40, 42, 53, and 54.

CAMPELLO, R. J. G. B.; MOULAVI, D.; ZIMEK, A.; SANDER, J. A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. **Data Min. Knowl. Discov.**, v. 27, n. 3, p. 344–371, 2013. Citations on pages 34, 43, 45, 46, 85, and 86.

_____. Hierarchical density estimates for data clustering, visualization, and outlier detection. **ACM TKDD**, v. 10, n. 1, p. 1–51, 2015. Citations on pages 25, 38, 40, 42, 46, 52, 53, 54, 75, 85, 86, 87, 89, 90, and 101.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. Introduction to semi-supervised learning. In: CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. (Ed.). Cambridge, MA: MIT Press, 2006. chap. 1, p. 1–12. Citations on pages 23, 49, and 50.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine Learning**, v. 20, n. 3, p. 273–297, 1995. Available: <<https://doi.org/10.1007/BF00994018>>. Citation on page 107.

CRAENENDONCK, T. van; BLOCKEEL, H. Constraint-based clustering selection. **Machine Learning**, v. 106, n. 9–10, p. 1497–1521, 2017. Available: <<https://doi.org/10.1007/s10994-017-5643-7>>. Citation on page 46.

DAVIDSON, I.; WAGSTAFF, K. L.; BASU, S. Measuring constraint-set utility for partitional clustering algorithms. In: **PKDD06**. [S.l.: s.n.], 2006. p. 115–126. Citation on page 46.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **JMLR**, v. 7, p. 1–30, 2006. Citation on page 75.

DIGLES, D.; ECKER, G. F. Self-organizing maps for in silico screening and data visualization. **Molecular Informatics**, v. 30, n. 10, p. 838–846, 2011. Citation on page 63.

DUDA, R.; HART, P.; STORK, D. **Pattern classification (2nd Edition)**. [S.l.]: Wiley-Interscience, 2001. (Pattern Classification and Scene Analysis: Pattern Classification). Citation on page 105.

ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: **Proc. KDD**. [S.l.: s.n.], 1996. p. 226–231. Citations on pages 31, 35, and 52.

EVERIT, B.; LANDAU, S.; LEESSE, M. **Cluster Analysis**. [S.l.: s.n.], 2001. Citation on page 30.

FISCHER, E. Einfluss der configuration auf die wirkung der enzyme. **Berichte der deutschen chemischen Gesellschaft**, WILEY-VCH Verlag, v. 27, n. 3, p. 2985–2993, 1894. ISSN 1099–0682. Citation on page 62.

FONTAINE, F.; PASTOR, M.; ZAMORA, I.; SANZ, F. Anchor-grind: Filling the gap between standard 3d qsar and the grid-independent descriptors. **Journal of Medicinal Chemistry**, v. 48, n. 7, p. 2687–2694, 2005. PMID: 15801859. Available: <<https://doi.org/10.1021/jm049113+>>. Citations on pages 73 and 104.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. **J. Amer. Statist. Assoc.**, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937. Citation on page 75.

GAULTON, A.; HERSEY, A.; NOWOTKA, M.; BENTO, A. P.; CHAMBERS, J.; MENDEZ, D.; MUTOWO-MEULLENET, P.; ATKINSON, F.; BELLIS, L. J.; CIBRIÁN-UHALTE, E.; DAVIES, M.; DEDMAN, N.; KARLSSON, A.; MAGARIÑOS, M. P.; OVERINGTON, J. P.; PAPADATOS, G.; SMIT, I.; LEACH, A. R. The ChEMBL database in 2017. **Nucleic Acids Res.**, v. 45, n. Database-Issue, p. D945–D954, 2017. Citations on pages 73 and 104.

GERTRUDES, J. C. **Emprego de técnicas de análise exploratória de dados utilizados em Química Medicinal**. Master's Thesis (Master's Thesis) — Escola de Artes, Ciências e Humanidades - Universidade de São Paulo., 2013. Citations on pages 62, 63, and 64.

GERTRUDES, J. C.; MALTAROLLO, V. G.; SILVA, R. A. A.; OLIVEIRA, P. R.; HONÓRIO, K. M.; SILVA, A. B. F. D. Machine learning techniques and drug design. **Current Medicinal Chemistry**, v. 19, p. 4289–4297, 2012. Citations on pages 61, 62, and 64.

GEUSEBROEK, J.; BURGHOUTS, G. J.; SMEULDERS, A. W. M. The amsterdam library of object images. **International Journal of Computer Vision**, v. 61, n. 1, p. 103–112, 2005. Citation on page 89.

HAN, J. **Data Mining: Concepts and Techniques**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005. ISBN 1558609016. Citation on page 29.

HANDL, J.; KNOWLES, J. D. An evolutionary approach to multiobjective clustering. **IEEE Trans. Evolutionary Computation**, v. 11, n. 1, p. 56–76, 2007. Citations on pages 87 and 88.

HARTIGAN, J. A. **Clustering Algorithms**. [S.l.]: Wiley, 1975. ISBN 047135645X. Citations on pages 29, 31, and 36.

HASANLOEI, M. A. V.; SHEIKHPOUR, R.; SARRAM, M. A.; SHEIKHPOUR, E.; SHARIFI, H. A combined fisher and laplacian score for feature selection in qsar based drug design using compounds with known and unknown activities. **Journal of computer-aided molecular design**, Springer, v. 32, n. 2, p. 375–384, 2018. Citations on pages 64 and 123.

HINNEBURG, A.; KEIM, D. A. *et al.* An efficient approach to clustering in large multimedia databases with noise. In: **KDD**. [S.l.: s.n.], 1998. v. 98, p. 58–65. Citation on page 31.

HORTA, D.; CAMPELLO, R. J. G. B. Automatic aspect discrimination in data clustering. **Pattern Recognition**, v. 45, n. 12, p. 4370–4388, 2012. Citations on pages 88 and 89.

HUBERT, L.; ARABIE, P. Comparing partitions. **Journal of classification**, Springer, v. 2, n. 1, p. 193–218, 1985. Citation on page 87.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern Recognition Letters**, v. 31, p. 651–666, 2010. Citation on page 32.

JAIN, A. K.; DUBES, R. C. **Algorithms for Clustering Data**. [S.l.]: Prentice-Hall, 1988. Citations on pages 29, 30, 40, 45, and 87.

KAUFMAN, L.; ROUSSEEUW, P. J. **Finding groups in data: an introduction to cluster analysis**. [S.l.]: Wiley, 2005. Citations on pages 29 and 30.

KITCHEN, D. B.; STAHURA, F. L.; BAJORATH, J. Computational techniques for diversity analysis and compound classification. **Mini Reviews in Medicinal Chemistry**, v. 4, p. 1029–1039, 2004. Citation on page 62.

KONDRATOVICH, E.; BASKIN, I. I.; VARNEK, A. Transductive support vector machines: Promising approach to model small and unbalanced datasets. **Molecular Informatics**, Wiley Online Library, v. 32, n. 3, p. 261–266, 2013. Citation on page 64.

KRIEGEL, H.; SCHUBERT, E.; ZIMEK, A. The (black) art of runtime evaluation: Are we comparing algorithms or implementations? **Knowl. Inf. Syst.**, v. 52, n. 2, p. 341–378, 2017. Citation on page 78.

KRIEGEL, H.-P.; KRÖGER, P.; SANDER, J.; ZIMEK, A. Density-based clustering. **WIREs: Data Mining and Knowledge Discovery**, v. 1, n. 3, p. 231–240, 2011. Citations on pages 24, 31, 35, 50, and 52.

LELIS, L.; SANDER, J. Semi-supervised density-based clustering. In: **Proc. IEEE ICDM**. [S.l.: s.n.], 2009. p. 842–847. Citations on pages 52, 53, 54, 75, 90, and 91.

LI, J.; SANDER, J.; CAMPELLO, R. J. G. B.; ZIMEK, A. Active learning strategies for semi-supervised DBSCAN. In: **Proc. Canadian AI**. [S.l.: s.n.], 2014. p. 179–190. Citation on page 52.

LICHMAN, M. **UCI Machine Learning Repository**. 2013. Available: <<http://archive.ics.uci.edu/ml>>. Citations on pages 73, 74, 76, 88, and 89.

LIU, W.; CHANG, S. Robust multi-class transductive learning with graphs. In: **Proc. IEEE CVPR**. [S.l.: s.n.], 2009. p. 381–388. Citations on pages 52 and 74.

MACQUEEN, J. B. Some methods for classification and analysis of multivariate observations. In: CAM, L. M. L.; NEYMAN, J. (Ed.). **Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.]: University of California Press, 1967. v. 1, p. 281–297. Citation on page 32.

MALTAROLLO, V. G. **Relação Quantitativa entre Estrutura Química e Atividade Biológica de Substâncias com Afinidade pelo Receptor PPAR δ** . Master's Thesis (Master's Thesis) — Universidade Federal do ABC, 2009. Citations on pages 62 and 64.

MALTAROLLO, V. G. **Aplicação de estratégias in silico para o desenvolvimento de ligantes com afinidade pelo receptor PPAR δ** . Phd Thesis (PhD Thesis) — Universidade Federal do ABC, 2013. Citations on pages 25, 61, 62, 104, and 105.

MASAND, V. H.; RASTIJA, V. Pydescriptor: A new pymol plugin for calculating thousands of easily understandable molecular descriptors. **Chemometrics and Intelligent Laboratory Systems**, v. 169, p. 12 – 18, 2017. ISSN 0169-7439. Available: <<http://www.sciencedirect.com/science/article/pii/S016974391730312X>>. Citation on page 104.

MATLAB Optimization Toolbox. 2011. The MathWorks, Natick, MA, USA. Citation on page 76.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw Hill, 1997. Citation on page 29.

MOULAVI, D. **Finding, Evaluating and Exploring Clustering Alternatives Unsupervised and Semi-supervised**. Phd Thesis (PhD Thesis) — University of Alberta, 2014. Citation on page 42.

NALDI, M. C.; CAMPELLO, R. J. G. B.; HRUSCHKA, E. R.; CARVALHO, A. C. P. L. F. Efficiency issues of evolutionary k-means. **AppliedSoft Computing**, v. 11, n. 2, p. 1938–1952, 2011. Citations on pages 88 and 89.

NEMENYI, P. Distribution-free multiple comparisons. In: **Biometrics**. [S.l.: s.n.], 1962. v. 18, p. 263. Citation on page 75.

NING, X.; KARYPIS, G. In silico structure-activity-relationship (sar) models from machine learning: a review. **Drug Development Research**, Wiley Subscription Services, Inc., A Wiley Company, v. 72, n. 2, p. 138–146, 2011. Citation on page 63.

POURRAJABI, M.; MOULAVI, D.; CAMPELLO, R. J. G. B.; ZIMEK, A.; SANDER, J.; GOEBEL, R. Model selection for semi-supervised clustering. In: **Proc. EDBT**. [S.l.: s.n.], 2014. p. 331–342. Citation on page 90.

RIVERA-BORROTO, O. M.; PONCE, Y. M.; VEGA, J. M. Garcia-de-la; GRAU-ÁLBALO, R. d. C. Comparison of combinatorial clustering methods on pharmacological data sets represented by machine learning–selected real molecular descriptors. **J. Chem. Inf. Model.**, v. 51, p. 3036–3049, 2011. Citations on pages 74 and 104.

RUIZ, C.; SPILIOPOULOU, M.; MENASALVAS, E. Density-based semi-supervised clustering. **DataMinKnowlDisc**, v. 21, n. 3, p. 345–370, 2010. Citation on page 53.

RUIZ, C.; SPILIOPOULOU, M.; RUIZ, E. M. C-DBSCAN: density-based clustering with constraints. In: **Proc. RSFDGrC**. [S.l.: s.n.], 2007. p. 216–223. Citation on page 53.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Inf Process Manag**, v. 45, n. 4, p. 427–437, 2009. Citation on page 73.

SOUSA, C. A. R. de. An overview on the gaussian fields and harmonic functions method for semi-supervised learning. In: **Proc. IJCNN**. [S.l.: s.n.], 2015. p. 1–8. Citations on pages 50 and 52.

SOUSA, C. A. R. de; REZENDE, S. O.; BATISTA, G. E. A. P. A. Influence of graph construction on semi-supervised learning. In: **Proc. ECML PKDD**. [S.l.: s.n.], 2013. p. III. 160–175. Citations on pages 24, 50, 51, 74, and 75.

SOUTO, M. C. P. de; COSTA, I. G.; ARAUJO, D. S. A. de; LUDERMIR, T. B.; SCHLIEP, A. Clustering cancer gene expression data: a comparative study. **BMC Bioinformatics**, v. 9, 2008. Citations on pages 73 and 88.

SUTHERLAND, J. J.; O'BRIEN, L. A.; WEAVER, D. F. A comparison of methods for modeling quantitative structure-activity relationships. **J. Med.Chem.**, v. 47, n. 22, p. 5541–5554, 2004. Citations on pages 73 and 104.

SZUMMER, M.; JAAKKOLA, T. S. Information regularization with partially labeled data. In: **Proc. NIPS**. [S.l.: s.n.], 2002. p. 1025–1032. Citation on page 51.

TAN, C.; QIN, X.; LI, M. An ensemble method based on a self-organizing map for near-infrared spectral calibration of complex beverage samples. **Analytical and bioanalytical chemistry**, Springer, v. 392, n. 3, p. 515–521, 2008. Citation on page 35.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. [S.l.]: Addison–Wesley, 2006. Citation on page 29.

THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern Recognition, Fourth edition**. London: [s.n.], 2009. 961 p. Citation on page 29.

TUCCINARDI, T. Docking-based virtual screening: Recent developments. **Combinatorial Chemistry & High Throughput Screening**, v. 12, p. 303–314, 2009. Citation on page 62.

VANSCHOREN, J.; RIJN, J. N. V.; BISCHL, B.; TORGO, L. Openml: networked science in machine learning. **ACM SIGKDD Expl. Newsletter**, v. 15, n. 2, p. 49–60, 2014. Citations on pages 73 and 88.

VARNEK, A.; BASKIN, I. Machine learning methods for property prediction in chemoinformatics: Quo vadis? **Journal of Chemical Information and Modeling**, v. 52, n. 6, p. 1413–1437, 2012. Citation on page 63.

XU, R.; Wunsch-II, D. C. **Clustering**. New Jersey: [s.n.], 2009. 358 p. Citations on pages 29 and 30.

YEUNG, K. Y.; FRALEY, C.; MURUA, A.; RAFTERY, A. E.; RUZZO, W. L. Model-based clustering and data transformations for gene expression data. **Bioinformatics**, v. 17, n. 10, p. 977–987, 2001. Citations on pages 88 and 89.

YEUNG, K. Y.; MEDVEDOVIC, M.; BUMGARNER, R. E. Clustering gene-expression data with repeated measurements. **Genome Biol**, v. 4, n. 5, p. R34, 2003. Citations on pages 73 and 88.

YOUNG, D. C. **Computational drug design– A Guide for Computational and Medicinal Chemists**. [S.l.]: Wiley, 2009. Citations on pages 25 and 61.

ZHAO, J.; LU, K.; HE, X. Locality sensitive semi-supervised feature selection. **Neurocomputing**, v. 71, n. 10-12, p. 1842–1849, 2008. Available: <<https://doi.org/10.1016/j.neucom.2007.06.014>>. Citations on pages 106 and 107.

ZHAO, L.; LUO, S.; TIAN, M.; SHAO, C.; MA, H. Combining label information and neighborhood graph for semi-supervised learning. In: **Proc. ISNN – Advances in Neural Networks**. [S.l.: s.n.], 2006. p. 482–488. Citation on page 51.

ZHU, X. **Semi-Supervised Learning Literature Survey — TR1530**. [S.l.], 2005. Citations on pages 24 and 50.

ZHU, X.; GHAHRAMANI, Z.; LAFFERTY, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In: **Proc. ICML**. [S.l.: s.n.], 2003. p. 912–919. Citations on pages 51 and 74.

ZHU, X.; GOLDBERG, A. B. **Introduction to Semi-Supervised Learning**. [S.l.]: Morgan & Claypool Publishers, 2009. (Synthesis Lectures on Artificial Intelligence and Machine Learning). Citations on pages [23](#), [49](#), [50](#), and [52](#).