Anytime discovery of a diverse set of patterns with Monte Carlo tree search

Guillaume Bosc

N°d'ordre NNT : 2017LYSEI074

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
l'INSA de Lyon

Ecole Doctorale N° 512
Mathématiques et Informatique (InfoMaths)

Spécialité/ discipline de doctorat : Informatique

Soutenue publiquement le 11/09/2017, par :
Guillaume Bosc

Anytime discovery of a diverse set of patterns with Monte Carlo tree search

---

Devant le jury composé de :

Amer-Yahia, Sihem	Directrice de recherche, CNRS	Présidente
Calders, Toon	Professeur, Univesiteit Antwerpe	Rapporteur
Cazenave, Tristan	Professeur, Université Paris-Dauphine	Rapporteur
Bensafi, Moustafa	Directeur de recherche, CNRS	Examinateur
Flach, Peter	Professeur, University Bristol	Examinateur
Morik, Katharina	Professeure, Technische Universität Dortmund	Examinatrice
Boulicaut, Jean-François	Professeur, INSA Lyon	Directeur de thèse
Kaytoue, Mehdi	Maître de conférences, INSA Lyon	Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec : Renée EL MELHEM Bat Blaise Pascal 3e etage secretariat@edchimie-lyon.fr Insa : R. GOURDON	M. Stéphane DANIELE Institut de Recherches sur la Catalyse et l'Environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 avenue Albert Einstein 69626 Villeurbanne cedex directeur@edchimie-lyon.fr
E.E.A.	**ELECTRONIQUE, ELECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec : M.C. HAVGOUDOUKIAN Ecole-Doctorale.eea@ec-lyon.fr	M. Gérard SCORLETTI Ecole Centrale de Lyon 36 avenue Guy de Collongue 69134 ECULLY Tél : 04.72.18 60.97 Fax : 04 78 43 37 17 Gerard.scorletti@ec-lyon.fr
E2M2	**EVOLUTION, ECOSYSTEME, MICROBIOLOGIE, MODELISATION** **http://e2m2.universite-lyon.fr** Sec : Sylvie ROBERJOT Bât Atrium - UCB Lyon 1 04.72.44.83.62 Insa : H. CHARLES secretariat.e2m2@univ-lyon1.fr	M. Fabrice CORDEY CNRS UMR 5276 Lab. de géologie de Lyon Université Claude Bernard Lyon 1 Bât Géode 2 rue Raphaël Dubois 69622 VILLEURBANNE Cédex Tél : 06.07.53.89.13 cordey@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTE** **http://www.ediss-lyon.fr** Sec : Sylvie ROBERJOT Bât Atrium - UCB Lyon 1 04.72.44.83.62 Insa : M. LAGARDE secretariat.ediss@univ-lyon1.fr	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 avenue Jean Capelle INSA de Lyon 696621 Villeurbanne Tél : 04.72.68.49.09 Fax :04 72 68 49 16 Emmanuelle.canet@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHEMATIQUES** **http://infomaths.univ-lyon1.fr** Sec :Renée EL MELHEM Bat Blaise Pascal, 3e étage Tél : 04.72. 43. 80. 46 Fax : 04.72.43.16.87 infomaths@univ-lyon1.fr	M. Luca ZAMBONI Bâtiment Braconnier 43 Boulevard du 11 novembre 1918 69622 VILLEURBANNE Cedex Tél :04 26 23 45 52 zamboni@maths.univ-lyon1.fr
Matériaux	**MATERIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec : Marion COMBE Tél:04-72-43-71-70 –Fax : 87.12 Bat. Direction ed.materiaux@insa-lyon.fr	M. Jean-Yves BUFFIERE INSA de Lyon MATEIS Bâtiment Saint Exupéry 7 avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72.43 71.70 Fax 04 72 43 85 28 Ed.materiaux@insa-lyon.fr
MEGA	**MECANIQUE, ENERGETIQUE, GENIE CIVIL, ACOUSTIQUE** **http://mega.universite-lyon.fr** Sec : Marion COMBE Tél:04-72-43-71-70 –Fax : 87.12 Bat. Direction mega@insa-lyon.fr	M. Philippe BOISSE INSA de Lyon Laboratoire LAMCOS Bâtiment Jacquard 25 bis avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72 .43.71.70 Fax : 04 72 43 72 37 Philippe.boisse@insa-lyon.fr
ScSo	**ScSo*** **http://recherche.univ-lyon2.fr/scso/** Sec : Viviane POLSINELLI Brigitte DUBOIS Insa : J.Y. TOUSSAINT Tél : 04 78 69 72 76 viviane.polsinelli@univ-lyon2.fr	M. Christian MONTES Université Lyon 2 86 rue Pasteur 69365 LYON Cedex 07 Christian.montes@univ-lyon2.fr

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Remerciements

En premier lieu, je tiens sincèrement à remercier Toon Calders, professeur à l'Univesiteit Antwerpe et Tristan Cazenave, professeur à l'Université Paris-Dauphine, d'avoir accepté d'être rapporteurs de ce manuscrit de thèse. Je les remercie particulièrement pour le travail considérable qu'ils ont consacré à la lecture assidue de ce mémoire, malgré les délais serrés et leurs tâches professionnelles respectives.

Je remercie également Sihem Amer-Yahia, directrice de recherche CNRS, Moustafa Bensafi, directeur de recherche CNRS, Peter Flach, professeur à l'University Bristol, et Katharina Morik, professeure à la Technische Universität Dortmund, de m'avoir fait l'honneur de participer au jury de thèse. Je les remercie pour l'intérêt qu'ils ont porté à mon travail ainsi que pour les échanges pertinents qui ont suivi la soutenance.

J'adresse également de très chaleureux remerciements à Jean-François Boulicaut, professeur à l'INSA de Lyon, et Mehdi Kaytoue, maître de conférences à l'INSA de Lyon, pour la confiance qu'ils m'ont accordée ainsi que pour l'investissement dont ils ont fait preuve tout au long de la préparation de ma thèse. A leurs côtés, j'ai pu apprendre à la fois les règles de la recherche, mais également le goût prononcé de la rigueur et de la concision qu'exige un bon travail scientifique. Pour toutes ces discussions à la fois riches et constructives que nous avons eues, je ne peux que les remercier profondément. Au-delà d'être deux scientifiques compétents, j'ai rencontré deux personnalités marquantes qui ne pourront que m'inspirer pour la suite.

Durant ces années de thèse, j'ai également eu l'immense fierté d'avoir pu collaborer avec Moustafa Bensafi, directeur de recherche CNRS. Nos différents échanges m'ont permis de toujours garder à l'esprit que l'informatique, et particulièrement la fouille de données, reste avant tout un outil destiné à un utilisateur final. Aussi, je le remercie pour le temps et l'énergie qu'il a consacrés à mon travail afin qu'il soit le plus en adéquation possible avec les exigences requises en neurosciences.

Lors de la préparation de cette thèse, j'ai pu évoluer au sein d'un laboratoire de qualité. Je remercie en particulier les membres de l'équipe DM2L avec lesquels j'ai eu la chance de travailler, Marc, Céline et Pierre ainsi que mes collègues doctorants que j'ai rencontrés et avec qui j'ai partagé ces trois belles années. La préparation de cette thèse m'a également permis de faire de belles rencontres lors de conférences ou autres évènements scientifiques : il serait trop long de les nommer, mais merci à toutes et à tous !

Enfin, je tiens à remercier très sincèrement mes proches, mes amis et ma famille. Merci pour vos encouragements, votre soutien et votre bienveillance afin que je mène à bien ce beau et long projet qu'est la préparation d'un doctorat. Je remercie en particulier Lucie qui m'a soutenu jour après jour et qui a su trouver les mots pour me motiver durant les moments de doute. Merci également à notre fils, Léandre, qui est venu illuminer les derniers mois de ma thèse par sa présence et son sourire.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Knowledge discovery in databases

With the exponentially growth of data in the world, the need to extract and derive knowledge from this so-called big data is more important than ever. Indeed, smartphones with a GPS tracker are carried in the pocket of billions of people around the world: Tens of billions of trajectories are available. Moreover, the marketing of smart watches makes available a huge amount of health data that are collected from the customers' wrist. In addition, there is a data explosion in science: Numerous scientific data are available in several domains. For example, in biology, several DNA databases have been released to gather nucleotide sequence data from all organisms; genome databases collect genome sequences; or other databases concern the protein structure models. In astronomy, there are databases that gather billions of images taken by telescopes such as Hubble Telescope. With the multiplication of online tools that provide data on almost every application domain, it is now possible to get large amount of data. For instance, one can crawl thousands of physico-chemical properties for tens of thousands of molecules such as the molecular weight, the number of carbon atoms or the unsaturation index.

The goal is now to analyze this available data to elicit some knowledge. For instance, from the images taken by telescopes in the astronomy domain, some planets have been found and the scientists suggest, based on the data they collected, that they are possibly habitable. Deriving knowledge from raw data is a difficult task. For that, a process called Knowledge Discovery in Databases (KDD) has been proposed. This process aims to extract some knowledge from the data that is non trivial, new and actionable by the experts of the domain [34, 67]. Even if this KDD process was defined twenty years ago, it remains relevant to this day.

The KDD process is a user-centered approach. It is composed of five steps, from the selection and the preparation of the data to the interpretation of the results after a data mining task (see Figure 1). This process was guided by the need to analyze large amount of data that was initially studied manually by the experts of the domain. The goal is to transform a raw and low-level data into other forms that might be cleaner and easier to study. The study is performed thanks to the data mining step that is the heart of the KDD process, and the result of this step is visualized and interpreted by the experts to highlight hypotheses and, hopefully, to draw tangible conclusions. Basically, each of the five steps can be defined as follows:

- **Selection**: This step consists in selecting the data in which we are interested from the raw data that has been collected. The selected data is called the target data. For example, one can select the period of interest, a geographical area or a specific market to analyzed the purchased products.

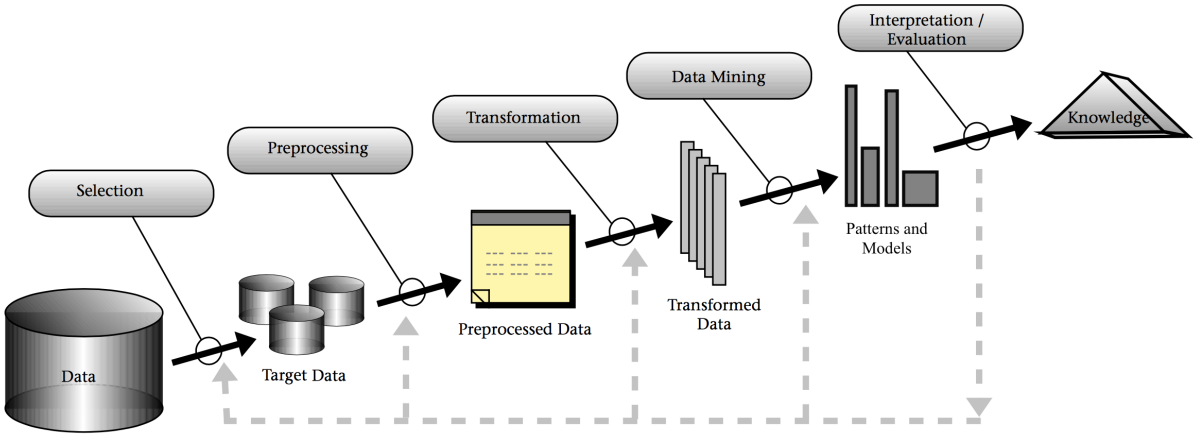


Figure 1: An overview of the KDD process (source: [67]).

- **Preprocessing**: This step aims at cleaning and preparing the target data. Missing values can be handled in different ways, such as predicting it with data mining tools. Besides we can cross-check when combining or summarising data from different sources to avoid misinterpretation. Handling the noise within a data is a common task. For instance, in the images taken by telescopes, it is required to reduce the light noise that can bias the interpretation.

- **Transformation**: Once the data is preprocessed, sometimes, it is required to transform it to be used as the input of the data mining tool. Indeed, some algorithms require the data in a specified format. Besides, a data reduction can be performed to avoid correlated descriptors that are useless and redundant. Feature selection enables to reduce the number of descriptors to only keep the more relevant ones [78]. Moreover, when facing numerical descriptors, a discretization is often performed in pre-processing task to turn them nominal.

- **Data mining**: This is the heart of the KDD process. From the transformed data, an algorithm explores the data and extract interesting patterns from it. There exists several data mining tasks, such as classification that builds a global model on the data or pattern mining that aims to exhibit local models. Also, one can be interested in outliers detection to identify anomalous behavior in the data. In addition, clustering is a popular method that aims to create so-called clusters of objects [18]. Objects in the same cluster are more similar (belonging to the same class) to each other than to those in other clusters.

- **Interpretation/Evaluation**: This is the last but not the least step of the KDD process. From the patterns extracted with the data mining method, it aims at deriving knowledge that is non trivial and actionable by the experts of the domain. For that, the visualization tools are very useful to understand the results.

Within an inter-disciplinary project, we have worked in collaboration with several neuroscientists and chemists to elicit new hypotheses on the structure-odor relationship (SOR) in olfaction. Our methodology follows the KDD process to derive some new knowledge on the SOR problem from raw data collected by the neuroscientists.

1.2 Application in neuroscience to elicit new hypotheses on the olfactory percept

Around the turn of the century, the idea that modern, civilized human beings might do without being affected by odorant chemicals became outdated: The hidden, inarticulate sense associated with their perception, hitherto considered superfluous to cognition, became a focus of study in its own right and thus the subject of new knowledge. It was acknowledged as an object of science by Nobel prizes (e.g., [37] awarded 2004 Nobel prize in Physiology or Medicine); but also society as a whole was becoming more hedonistic, and hence more attentive to the emotional effects of odors. Odors are present in our food, which is a source of both pleasure and social bonding; they also influence our relations with others in general and with our children in particular. The olfactory percept encoded in odorant chemicals contribute to our emotional balance and wellbeing: Olfactory impairment jeopardizes this equilibrium.

Neuroscience studies revealed that odor perception results from a complex phenomenon rooted in the chemical properties of a volatile molecule (described by multiple physicochemical descriptors) further detected by our olfactory receptors in the nasal cavity. Indeed, Figure 2 illustrates the process of the olfactory system: When someone smells a flower, the volatile molecules of the flower come to get binding in the olfactory receptors in the nasal cavity. From each receptor, a neural signal is then transmitted to central olfactory brain structures, i.e, the olfactory bulb [85]. At this stage, the combination of all these signals generates a complete neural representation, called "odor" that can then be described semantically by various types of perceptual quality (e.g., fruity, floral or woody).

While it is generally agreed that the physicochemical characteristics of odorants affect the olfactory percept, no simple and/or universal rule governing this Structure Odor Relationship (SOR) has yet been identified. Why does this odorant smell of roses and that one of lemon? Considering that the totality of the odorant message was encoded within the chemical structure, chemists have tried for a long time to identify relationships between chemical properties and odors. Topological descriptors, eventually associated with electronic properties or molecular flexibility were tentatively connected to odorants descriptors. However, it is now quite well acknowledged that structure-odor relationships are not bijective, meaning that the chemical space and the perceptual space connections are more subtle than previously thought[1]. For example, very different chemicals trigger a typical "camphor" smell, while a single molecule, the so-called "cat-ketone" odorant, elicit two totally different smells as a function of its concentration [50]. At best, such SOR rules are obtained for a very tiny fraction of the chemical space, emphasizing that they must be decomposed into sub-rules associated with given molecular topologies [55]. As such, this lack of bijective relationship must be handled. It suggests that a simple, universal and perfect rule does probably not exist, but instead, a combination of several sub-rules should be put forward to encompass the complexity of SOR.

During the preparation of this PhD., we applied a data science approach with a view to advancing the state of the art in understanding the mechanisms of olfaction. We created an interdisciplinary synergy between neuroscientists, chemists and data miners to support the emergence of new hypotheses[2]. For that, we applied the KDD process, from data collection to the discovery of new hypotheses on SOR. Basically, the neuroscientists gathered the data from two different

[1]There is a strong inter- and intra-individual variability when individuals are asked about the quality of an odor. There are several explanations: Geographical and cultural origins, each individual repertory of qualities (linguistic), genetic differences (determining olfactory receptors), troubles such as *anosmia* (see [86, 41]).

[2]This collaboration is partially supported by the CNRS (Préfute PEPS FASCIDO and the project "Mission pour l'interdisciplinarité") and the Institut rhônalpin des systèmes complexes (IXXI).

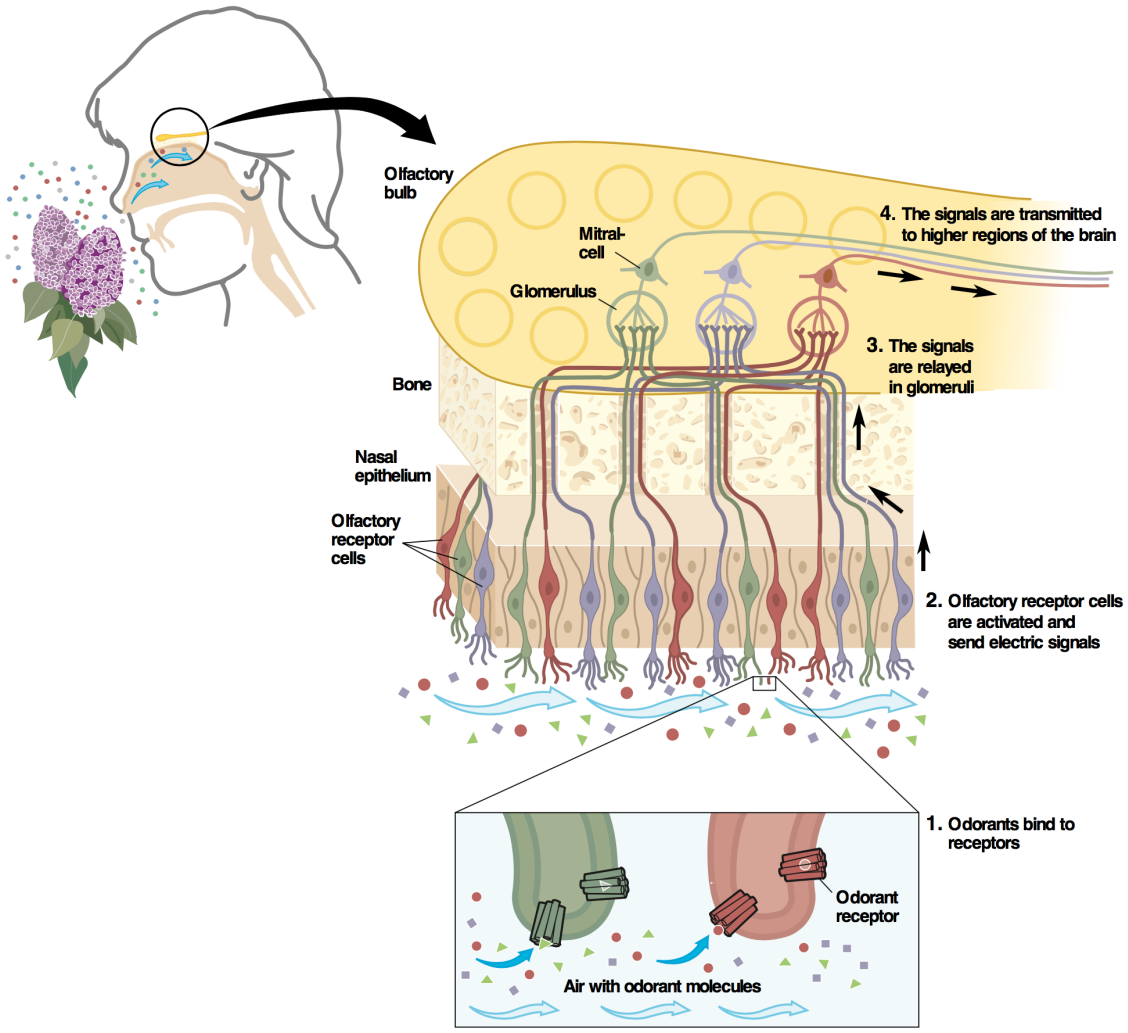
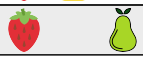


Figure 2: The olfactory system (source: [37]).

sources: (i) The *Dragon 6* software (available on `talete.mi.it`), and (ii) the well known atlas Arctander [9]. The first one provides the values of almost $5,000$ physico-chemicals properties (e.g., the *molecular weight* or the *number of carbon atoms*) for thousands of chemical molecules. The latter is an atlas created by scent experts that maps $1,689$ molecules to subset of odors among 74 olfactory qualities such as *Fruity*, *Vanilin*, or *Camphor*. From these two raw data, the neuroscientists pre-processed the data to keep only the molecules that are present in both of them. Moreover, the chemists identified and pointed out a set of 82 physico-chemical properties that have been known or guessed to be discriminant in SOR. Thus, the final data is made of $1,689$ molecules described by 82 physico-chemical properties and associated to an average of 2.88 odors among the 74 olfactory qualities.

Data mining methods can be applied on this transformed data to address the SOR discovery problem, either through the building of predictive models or through rules discovery. One obstacle to this is that olfactory datasets are "complex" (i.e., several thousand of dimensions, heterogeneous descriptors, multi-label, imbalanced classes, and non robust labelling) and, above all a lack of data-centric methods in neuroscience suitable for this level of complexity. The main aim of our study is to examine this issue by linking the multiple molecular characteristics of odorant

Table 1: Example of an olfactory dataset.

ID	MW	nAT	nC	Odor
1	150	21	11	
24	128	29	9	
48	136	24	10	
60	152	23	11	
82	151	27	12	
1633	142	27	10	

molecule to olfactory qualities (fruity, floral, woody, etc.) using a descriptive approach (pattern mining). Indeed, a crowd-sourced challenge was recently proposed by *IBM Research* and *Sage* called *DREAM Olfaction Prediction Challenge* [65]. The challenge resulted in several models that were able to predict especially pleasantness and intensity and 8 out of 19 olfactory qualities in their dataset ("garlic", "fish", "sweet", "fruit", "burnt", "spices", "flower" and "sour") with an average correlation of predictions across all models above 0.5 [91]. Although this is a clinical data (the perception is not given by scent experts), these findings are timely and interesting because they show the existence of a predictive relationship between some olfactory qualities and the physico-chemical properties of odorant molecules. Nevertheless, to go further into the understanding of the stimulus-percept issue in olfaction, it is important to explain and thus to isolate physico-chemical descriptive rules allowing describing these olfactory qualities (based on scent experts' perception). Identifying such physico-chemical rules characterizing odor quality opens a hitherto little explored field in neuroscience research, shedding new lights on olfactory system functioning.

Example 1. *Let us consider the example of an olfactory dataset given in Table 1. This data concerns 6 molecules given by their ID. Each molecule is described by 3 physico-chemical properties: The molecular weight MW, the number of atoms nAT, and the number of carbon atoms. Besides, each molecule is associated to several odors among strawberry, honey and pear. For instance the scent experts state that Molecule* 24 *smells honey and pear. From this data, the aim is to identify so-called descriptive rules on the physico-chemical properties that are characteristic of odors. For instance, let us consider the rule* $\langle [128 \leqslant MW \leqslant 151], [23 \leqslant nAT \leqslant 29] \rangle \rightarrow \{Honey\}$. *The Molecules 24, 48, 82 and 1633 support the body of this rule (i.e.,* $\langle [128 \leqslant MW \leqslant 151], [23 \leqslant nAT \leqslant 29] \rangle$*). Thus,* 75% *of the molecules verifying the body of this rule smell* honey*, whereas* 50% *of the molecules in the data are associated to* honey*. We can state that this rule is interesting since it means that molecules with a molecular weight below* 151 *and with at least* 23 *atoms are more likely to smell* honey*. Finding such a rule allows to derive some hypotheses and, maybe, new knowledge on the structure-odor relationship.*

Thus, the extraction of rules on SOR in such an olfactory dataset raises several questions. In the next section, we give an overview on these problems and how we resolve them through our two main contributions.

Table 2: Market basket data.

TID	Products
1	
2	
3	
4	
5	
6	

Table 3: Labeled version of Table 2.

TID	Products	Gender	Age range
1			
2			
3			
4			
5			
6			

1.3 Supervised descriptive rule discovery

The discovery of hypotheses on the structure-odor relationship is an interesting application for data mining. Indeed, the complex structure of the data combined with the specific goals of the neuroscientists and chemists, highlight several opened issues that have to be solved. In this section, we do not consider the olfactory data but we illustrate pattern mining within a simpler setting: Market basket analysis. Pattern mining has been deeply studied in the three last decades [74]. Indeed, it is now commonly taught in textbooks and applied in the industry [133, 144, 3]. The first works on pattern mining dealt with the extraction of frequent itemsets [4] to discover some association rules between itemsets [4]. An itemset is a set of items, i.e., properties of some objects. For instance, a well known application of association rules concerns the market basket analysis: The aim is to extract rules between products of a supermarket that are often bought together by customers. Table 2 is a toy market data: Each line is about the products purchased by a customer. Note that a product is an item, and a set of products forms a so-called itemset. For example, the second customer has bought beers, vegetables and diapers. A popular anecdote about unexpected association rules states that the customers buying diapers tend also to buy beers. In Table 2, the confidence of the association rule $Diapers \rightarrow Beers$ is 0.75 since three of the four customers who purchased diapers also bought beers. Thus, from everyday data, some association rules are identified and can then be derived into actionable knowledge: A supermarket can either put diapers close to the beers to reduce the time spent by the customers, or to position them diagonally opposite to force the customer to pass through the supermarket shelves to make him buy more products.

Initially, the market basket data is not labeled: Each line of the data gathers the products bought by a customer. However, more and more applications in pattern mining are based on labeled data. In a labeled version of this market basket data, each of the customers can also be associated to a label, for example *male* or *female*. Table 3 illustrates the labeled version of the market basket data of Table 2. Thus, we can exhibit some association rules with high confidence for women but that are not supported by men and vice versa: For instance, the confidence of the rule $Diapers \rightarrow Beers$ is 1 for men (i.e., all men who purchased diapers also bought beers) and 0.5 for women (i.e., half of the women who purchased diapers also bought beers). Alternatively, the labels can concern the age range of the customers (taking values in $\{[0, 29], [30, 59], [60, 99]\}$), and thus we can find some association rules for a specific age range that is no longer true for other

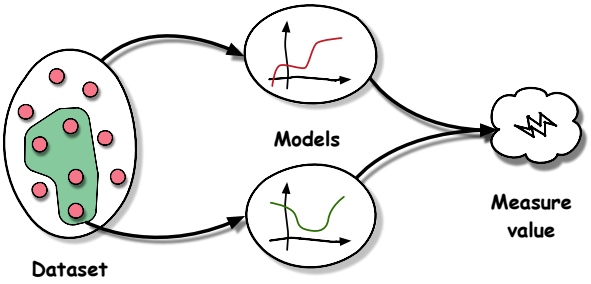


Figure 3: A typical EMM instance.

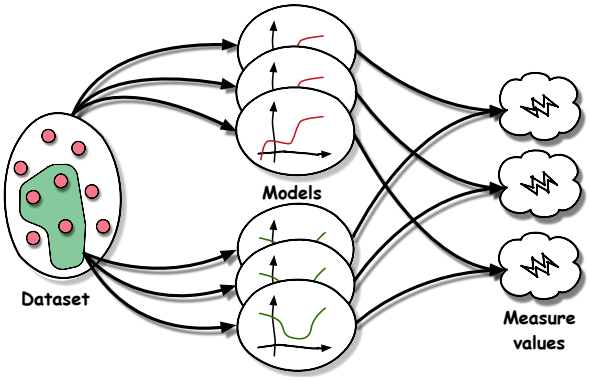


Figure 4: EMM considering target subspaces.

age ranges. It leads to the definition of several supervised data mining tasks. Generally, with labeled data, one is interested into learning a model that can state to which of the groups (also called classes) a new object (e.g., a customer or a molecular molecule) belongs: A typical machine learning task [141]. Support vector machine [81, 132] or deep learning [103] are techniques dealing with labeled data. They aim at building a classification model that can predict the label (or the class) of a new object. However, such efficient learning methods are black boxes: The user does not know why the algorithm predicts a specific label for an object. They do not allow to exhibit intelligible rules that can be interpreted and understood to be actionable. Note that some learning methods propose intelligible models, such as decision trees, but the model may be weak [127]. Finding such rules is possible with supervised descriptive rule discovery [119].

The discovery of descriptions which distinguish a group of objects given a target (class) has been widely studied in data mining and machine learning community under several terms (subgroup discovery [142], emerging patterns [56], contrast sets [15], hypotheses in formal concept analysis [70]) [119]. Let us now consider supervised descriptive rule discovery in the well-established framework of subgroup discovery (SD) [142]. Given a set of objects (e.g., customers) taking a vector of attributes (of boolean, nominal, or numerical type) as a description (e.g., purchased products), and a class label as a target (e.g., gender), the goal is to discover subgroups of objects for which there is a significant difference between the label distribution within the group compared to the distribution within the whole dataset, e.g. considering the difference of precision with the well-known weighted relative accuracy (WRAcc) [102]. In others terms, it searches for descriptive rules that conclude on a single label with possibly few errors.

However, many datasets exhibit objects associated to several labels. For example, in Table 3, each customer is associated to two labels: The gender and the age range. For these so-called multi-labeled data, SD has been generalized into the exceptional model mining (EMM) framework [104] to capture the more complex relationships that exist in the data between the descriptive attributes (e.g., the products) and the class labels (e.g., the gender of the customer). For that, the notion of *model* has been introduced. Each set of objects induces a specific model that is built on the class labels. Several models have been proposed in the state of the art [59], e.g., the linear regression between two numerical class attributes [58], the contingency tables for several class labels [114] or the Bayesian networks computed on the class labels [61]. The goal of EMM is to extract subgroups of objects for which the model built on the class labels is significantly different from the model induced by the entire dataset. Figure 3 displays a typical EMM instance. The data is displayed on the left. From the entire data, we can compute the model built on the class labels: This is the red curve. In this example, we can consider that the curve represents the power

Table 4: The contingency table of all the customers in Table 3.

	0+	30+	60+
♀	0.17	0	0.33
♂	0.17	0.33	0

Table 5: The contingency table of customers that purchased diapers and beers.

	0+	30+	60+
♀	0.33	0	0
♂	0	0.67	0

consumption of a population during one day: For the entire population, there is an increase of the power consumption between 7p.m. and 12p.m.. Then, the subset of objects (called the subgroup) in the green area is selected to be analyzed. We compute the model built on the class labels by this subgroup: This is the green curve. Then a quality measure is used to compare these two models. If the quality measure is high, it means that the models are significantly different: The subgroup of objects is said interesting because it behaves differently on the class labels than the entire set of objects. For instance, we have discovered that this subgroup concerns the night workers.

Example 2. *Let us consider Table 3. Each consumer is labeled with both its gender and its age range. We can use the model of the contingency tables on the two class attributes gender and age range given is Table 4 for all the customers. The subgroup given by the customers that purchased both beers and diapers contains three people. The contingency table induced by this subgroup is given in Table 5. The total variation distance is then used as the quality measure to compare the two contingency tables [114]. This distance sums the absolute difference between each pair of cells divided by 2. Thus, the quality measure is:* $\frac{|0.17-0.33|+|0-0|+|0.33-0|+|0.17-0|+|0.33-0.67|+|0-0|}{2} = 0.5$.

Example 3. *Consider the olfactory data in Table 1. Suppose that we use the probability distribution model. It consists to compare the frequency of a label in a subgroup to its frequency in the entire data. For that, we can use the mean of the WRAcc measure of the subgroup for each label [2]. First, the relative frequencies of the odors* strawberry, honey *and* pear *in the data are respectively* 0.67, 0.5 *and* 0.5. *Indeed, four molecules among the six molecules of the data are labeled as* strawberry. *Let us consider the subgroup containing the molecules with a molecular weight below* 151 *and with at least* 23 *atoms. This subgroup is given by the description* $\langle [128 \leqslant MW \leqslant 151], [23 \leqslant nAT \leqslant 29] \rangle$. *This subgroup contains the Molecules 24, 48, 82 and 1633. The relative frequencies of the odors* strawberry, honey *and* pear *in this subgroup are respectively* 0.5, 0.75 *and* 0.75. *The WRAcc measure of the subgroup for the* strawberry *is given by:* $\frac{4}{6} \times (0.5 - 0.67) = -0,11$. *Similarly, the WRAcc measure for* honey *is* 0.17 *and for* pear *is also* 0.17. *The mean of these 3 measures evaluate the exceptional behavior of the subgroup on the three odors: The quality measure of the subgroup is* 0.08.

In subgroup discovery and exceptional model mining, the enhancements proposed in the recent works can be divided into three categories: (i) The pattern language, (ii) the model with its quality measure and (iii) the algorithmic method to discover the best subgroups. The improvements on the pattern languages deal with the expressiveness of the patterns: From itemsets to numerical attributes, sequences [6], graphs [48] and so on. The enhancements on the quality measures consist in proposing new models for EMM to elicit different relationships in the data. The algorithmic improvements concern the methods designed to extract efficiently the best patterns

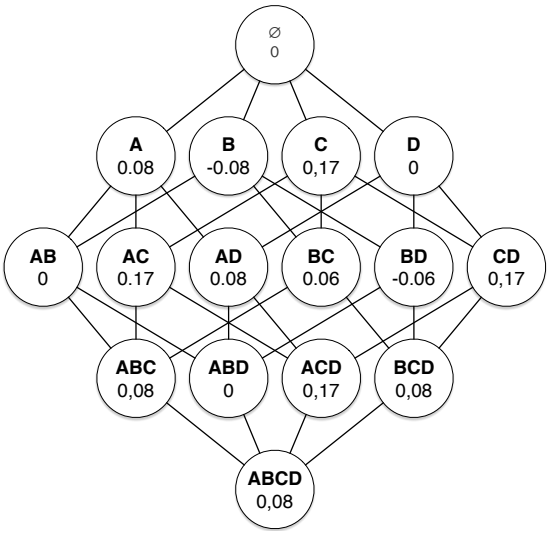


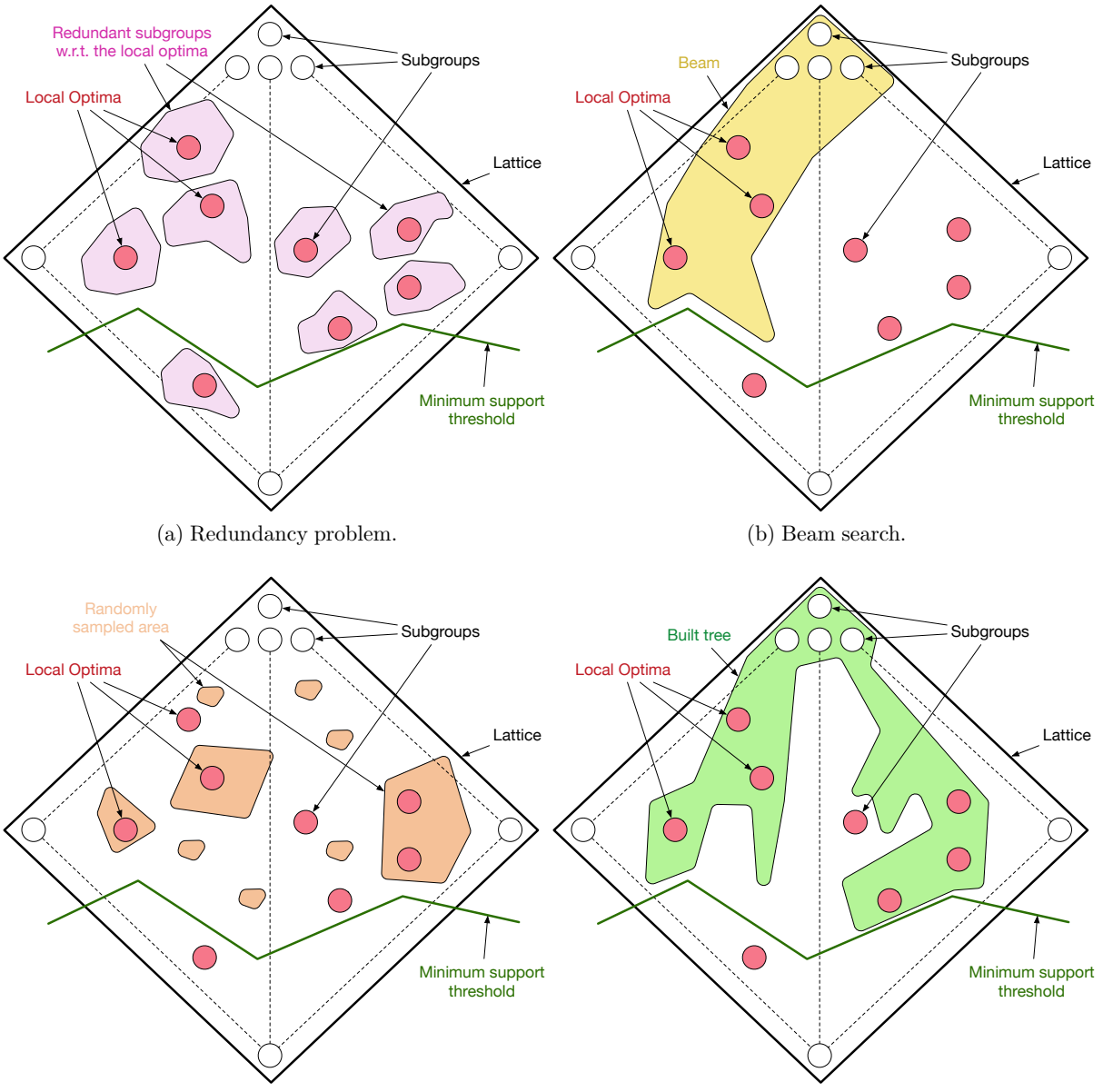
Figure 5: The lattice induced by Table 2 where A, B, C and D respectively correspond to the beers, the milk, the vegetable and the diapers. The value in the bottom of each node corresponds to the WRAcc measure of the itemset w.r.t. the male label.

w.r.t. the quality measure in an exhaustive or heuristic way. Nevertheless, there are still open issues that were highlighted when discussing our needs for olfactory data analysis.

1.4 Contributions

1.4.1 Monte Carlo tree search for pattern mining

In SD/EMM, the algorithms have to explore the set of subgroups, called the search space, to extract the best ones w.r.t. the quality measure, also called local optima. Basically, the search space contains all the possible subgroups defined by their description, i.e., a list of restrictions on the domain of the descriptive attributes (e.g., the physico-chemical properties). Note that a subgroup is either given by its description or by the set of objects it covers, called the extent. The search space is often a lattice since a partial order exists between the descriptions. Figure 5 represents the lattice of Table 2 where A, B, C and D respectively correspond to the beers, the milk, the vegetable and the diapers. The value in the bottom of each node corresponds to the WRAcc measure of the itemset w.r.t. the male label. Note that the WRAcc measure is not an increasing or decreasing function on the lattice: The WRAcc of a node (i.e., an itemset) can be more or less than those of its parents nodes. The itemsets with the maximum values of WRAcc, also called the local optima, are $\{C\}$, $\{A, C\}$, $\{C, D\}$ and $\{A, C, D\}$. In this thesis, we illustrate the lattice with the formalism of Figure 6a: The circles are the subgroups given by their description. The top subgroup is the more general subgroup: It contains all the objects of the data. The red circles are the local optima to find from this search space, i.e., the patterns that cover a set of objects that induces a model on the class labels significantly different from those induced by the entire data. The subgroups above the green curve are the subgroups containing at least $minSupp$ objects, where $minSupp$ is called the minimum support threshold. This constraint ensure to enumerate subgroups that are not too small, because subgroups containing few objects do not bring much information. The goal of SD/EMM is to extract the local optima that are above the green curve.



(a) Redundancy problem.

(b) Beam search.

(c) Sampling exploration.

(d) MCTS-based exploration.

Figure 6: Illustration of the different SD search algorithms.

The first algorithms proposed for SD were exhaustive, i.e., they enumerate all the possible descriptions by exploring the lattice of the subgroups [93, 12, 106]. Thus, exhaustive algorithms ensure to extract the local optima. However, they are not scalable for large data or for complex pattern languages (e.g., for numerical descriptive attributes[3], as is the case for the olfactory dataset). Indeed, the search space growths exponentially with the number of descriptive attributes and the number of objects. For that, recent works have proposed heuristic algorithms

[3]The only solution to handle numerical attributes is either to enumerate all the possible intervals [88] or to use a discretization in pre-processing task or to use on-the-fly discretization during the enumeration [66]. Note that most of existing works in SD/EMM (exhaustive methods included) employ a naive discretization as pre-processing task that converts numerical attributes into nominal one.

to explore the search space. Up to now, there exists three main heuristic methods for SD/EMM: (i) beam search [94, 138, 139, 140], (ii) evolutionary algorithms [54, 125] and (iii) sampling methods [114, 17]. Beam search is a greedy method that partially explores the search space with several hill climbings run in parallel [108]. It proceeds in a top down enumeration of the lattice, and it only explores a beam within the search space by keeping at each level the most promising subgroups. Figure 6b illustrates this heuristic strategy: It drives the exploration to the nearest best subgroups, ignoring lots of local optima. This strategy is the most popular one in SD/EMM [94, 138, 139, 140]. It quickly finds some (but not all) best subgroups. It generally works well when features have a very high discriminant power independently, i.e., when the local optima are "short" patterns. However, this method clearly lacks of exploration due to its greedy principle, especially when local optima are located everywhere in the search space. A second heuristic approach can exploit evolutionary algorithms. They use a fitness function to select which individuals to keep at the next generated population [14]. However, these methods need to set a lot of parameters to model the crossover rate, the size of the population, the process to update the generation, etc. Finally, contrary to the two previous heuristic approaches, sampling methods give a result anytime, i.e., a solution is always available. Sampling techniques are gaining interest because they allow direct interactions with the user for using his preferences to drive the search. However, traditional sampling methods used for pattern mining need a given probability distribution over the pattern space: This distribution depends on both the data and the quality measure [23, 114]. Each iteration is independent and consists of drawing a pattern given this probability distribution. Moreover, these probability distributions exhibit the problem of the long tail: There are many more uninteresting patterns than interesting ones. Thus, the probability to draw an uninteresting pattern is still high, and not all local optima may be drawn: There are no guarantee on the completeness of the result set. Figure 6c illustrates the principle of sampling methods: Several areas are sampled based on the probability distribution on the pattern space. Some of them contain local optima, but, due to the long tail problem, some sampled areas do not contain any local optimum.

With the so-called pattern explosion when considering frequent subgroups, the existing methods focus on top-k mining algorithms that aim at returning the k best encountered subgroups w.r.t. the quality measure. However, these approaches face the problem of redundancy: A subgroup with a slight change in its description gives a very similar subgroup and thus the quality measure of these subgroups are similar. In Figure 6a, in the area closed to the local optima, there are similar subgroups that are redundant with the local optimum. In top-k mining methods, a similarity measure is used as post-processing step to compare two subgroups to avoid the presence of redundant subgroups in the result set. Several similarity measures have been proposed, based on the similarity of the support (using the Jaccard coefficients) or on their description [138, 139].

Contribution 1:
As a first contribution, we propose a new sampling exploration method based on Monte Carlo tree search (MCTS) [95, 36]. It is an iterative method that employs random simulations to guide the exploration of the search space. This method exhibits several advantages w.r.t. existing heuristic approaches in SD/EMM. First, contrary to traditional pattern sampling methods, it does not require any probability distribution on the pattern space: The profile of the search space is learned incrementally. Second, it is based on the exploration / exploitation trade-off provided by the upper confidence bound [13]: This heuristic method aims at exploiting interesting solutions already found in the search space but also exploring few visited parts of the lattice. Besides, contrary to existing methods, we consider all the possible intervals for numerical attributes. Indeed, existing methods (exhaustive methods included) does not handle numerical descriptive attributes: They

proceed to a discretization (either as a pre-processing step or on-the-fly during the enumeration on the lattice) on the domain of these attributes to make them ordinal. In our approach, we avoid such a greedy method to handle numerical attributes and enumerate all the possible intervals [88] which is made possible with MCTS. Finally, this is an anytime pattern mining method that converges to the exhaustive search if given enough computational budget. Figure 6d illustrates the principle of using MCTS for SD/EMM: A search tree is incrementally built from the root of the lattice based on the trade-off between exploring rarely visited areas and exploiting interesting solutions. Given enough time, the result set will contain all the local optima.

1.4.2 Exceptional model mining in multi-label data

Exceptional model mining is the generalization of subgroup discovery that considers multi-label data [104]. Indeed, SD has been defined when each of the objects is associated to *one* label, e.g., in the market basket data a consumer is either a man or a woman. However, the real world data are often multi-label, e.g., an image can be labeled with several keywords (mountain, sunset, lake, etc.) or in the olfactory data each chemical molecule is labeled with several odors (fruity, spicy, vanilin, etc.). In this case, SD can not be applied and the EMM framework is required. Given a set of objects S, EMM consists in building the model induced by S on the class labels. In the recent works, several model classes have been proposed to exhibit the relationships in the data [59]. For example, let us consider the market basket data with two nominal target attributes given in Table 3: "Gender" which domain is $\{male, female\}$ and "Age group" taking value in $\{[0, 29], [30, 59], [60, 99]\}$. From this data, we used the contingency table between the two target attributes as a model (see Table 4 and Table 5). Then a quality measure is used to compare two contingency tables: The greater the quality measure, the more different the contingency tables. Thus, the goal of this EMM instance is to find out subgroups of customers whose contingency table is significantly different from the contingency table of all the customers in the data. In the olfactory data, we could have applied a model based on Bayesian networks on the odors [61]. Indeed, we compute the Bayesian network on the class labels (i.e., the odors) for a subset of molecules, and we use a quality measure to compare it with the Bayesian network on the odors for all the molecules in the data. Thus, a subgroup of molecules is said interesting if its Bayesian network is significantly different from those of the entire data. It means that in the subgroups the correlations between the labels are significantly different from the correlations between the labels in the entire data. Figure 3 illustrated the principle of an EMM instance: The model on the entire data is computed (red curve) and is compared to the model induced by a subset of objects (green curve) with a quality measure. The aim is to extract the set of the top-k subgroups w.r.t the quality measure that does not contain redundant subgroups.

However, for the SOR problem, the experts are not interested in model classes that are built on *all* the target labels. Indeed, finding a description about a subgroup of molecules that behaves differently from the entire data on *all* the odors is not actionable since each odor can have an impact. The neuroscientists and the chemists prefer to know that a subgroup of molecules whose model is different w.r.t. a (small) subset of odors. In this way, they can derive some knowledge about a specific subset of odors. This implies a substantial change in the definition of the EMM instance: A subgroup of objects no longer derives *one* model but *several* models, i.e., as many models as subsets of class labels. Figure 4 illustrates this change in the EMM setting. From a subset of objects, we can compute several models: One model for each subset of class labels. Thus, we compare each couple of models: For a given subset of class labels L, we compare the model built on L for the subgroup of objects and the model built on L for the entire data. The search space becomes even larger since we have to proceed to the Cartesian product between the

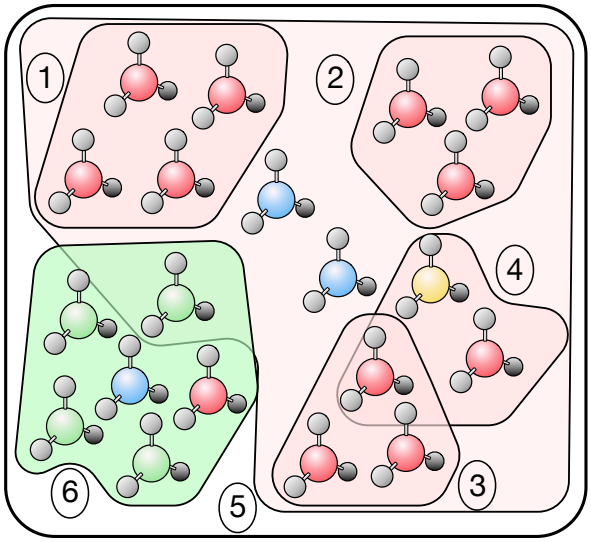


Figure 7: Example of groups of molecules based on their odor (represented by the color).

pattern search space and the power set of the class labels.

Besides, in the olfactory data, there is a high variance in the frequency of the class labels: Some odors such as fruity and floral are much more frequent than other odors such as musty and sandalwood. We say that a label is over-represented in the data if it is much more frequent than other labels. Moreover, the neuroscientists and the chemists require subgroups with as few errors as possible. It means that a subgroup is said interesting w.r.t. a subset of labels L if all the objects in the subgroup are labeled at least with L. In other words, the subgroup should not include objects that are not labeled with L in the data. This property is called the *precision* of the subgroup. In addition, the subgroup has to be as large as possible: It should contain lots of objects to be relevant. Indeed, it is easier to find small subgroups with a high precision than large subgroup with a good precision. This property is called the *recall* of the subgroup. Thus, the experts need to find subgroups for which the precision and the recall is high w.r.t. the subset of labels L. For example, in Figure 7, the red molecules (that smell *strawberry*) are over-represented in the dataset, but it is more interesting having the different subgroups (1), (2), (3) and (4) with high precision, rather than a single huge local subgroup (5) whose precision is much lower but with a good recall. Indeed, the experts of the domain suggest that a simple, universal and perfect rule does probably not exist, but instead, a combination of several sub-rules should be put forward to encompass the complexity of SOR. For molecules that are not over-represented, the extracted subgroups have to foster on both the precision and the recall: e.g., Subgroup (6) is interesting for the green molecules that smell *herbaceous*. Hence, we have to design a new model class that has to take into account: (i) The high variance in the frequency of the class labels, (ii) the precision and (iii) the recall of the subgroup w.r.t. the subset of class labels L.

Contribution 2:
As a second contribution, we propose a new EMM instance that deals with multi-label data. This EMM instance no longer derives *one* model by subgroups but several models: For each subgroup, as many models as subsets of labels are computed. Then the quality measure compare pairs of models that are built on the same subset of labels. Formally, if there are n different labels in the data, for each subgroup, we have to compute 2^n models, one for each subset of labels L.

Moreover, we propose a new model class called F_β that is based on the F_1 score. This model evaluates the precision and the recall of the subgroups w.r.t. a given subset of L. Besides, F_β includes a function $\beta(.)$ that takes into account the frequency of the subset of labels L, i.e., if L is over-represented in the data or not. For that, F_β fosters on the precision for over-represented labels and it converges to F_1 for other subsets of labels. In this way, we propose two quality measures that compare the model based on F_β for the subgroups and those of the entire data. This contribution, combined with the MCTS-based enumeration, enables to address the SOR problem, namely to extract diverse rules on physico-chemical properties that can explain the presence of diverse subsets of odors.

1.4.3 Application in neuroscience about the Structure-Odor relationship

Eliciting supervised descriptive rules on the structure-odor relationship to derive knowledge is a challenging task in neuroscience. Indeed, a crowd-sourced challenge was recently proposed by *IBM Research* and *Sage* called *DREAM Olfaction Prediction Challenge* [65]. The challenge resulted in several non intelligible models that were able to predict especially pleasantness and intensity for 8 out of 19 olfactory qualities in their dataset ("garlic", "fish", "sweet", "fruit", "burnt", "spices", "flower" and "sour") with an average correlation of predictions across all models above 0.5 [91]. However none descriptive rules have been proposed: The results only suggest that there exists links between some odors and the physico-chemical properties of the molecules. The current limited knowledge on SOR avoids the use of constraint-based pattern mining methods since setting a threshold remains hard. Indeed, the experts of the domain have difficulties to put words on their feelings. Thus, it is required to switch to other pattern mining methods that can handle the implicit expert's preferences. Interactive mining is a new trend of methods that includes the user in the exploration process [63]. It uses the user's preferences as a feedback to guide the exploration of the search space. Interactive mining methods rely on instant mining algorithms, i.e., algorithms that give results anytime. Beam search and sampling methods are instant mining approaches and can be tuned to interactive algorithms.

Contribution 3:
As a third contribution, we implement an interactive online application that allows the user to guide the exploration of the search space based on her preferences. In this application there are some visualization tools to present the results of the method. Based on this interactive platform, the neuroscientists and the chemists are working on the understanding of descriptive rules on the structure-odor relationship. It leads to a deep analysis of the rules, to elicit new hypotheses on the impact of some physico-chemical properties. Moreover, they are trying to combine several rules to improve the cover of a single rule. The results suggest promising perspectives for the understanding of the olfactory percept process.

1.5 Structure of the thesis

The thesis is organized as follows. Chapter 2 provides the background about pattern mining and formally defines subgroup discovery and exceptional model mining. In addition, we point out the main issues we propose to tackle in this thesis. Chapter 3 introduces the paradigm of Monte Carlo tree search, the exploration method we employ in our first contribution. We define its principle and its main enhancements proposed in the state of the art. Chapter 4 concerns our first contribution, namely applying MCTS for pattern mining. We formalize the problem and we propose a lot of strategies to adapt the traditional process of MCTS to SD/EMM. We thoroughly

experiment with benchmark datasets to assess the validity of this new exploration method. This contribution has been published in the proceedings of the French conference *Extraction et Gestion de Connaissances EGC'17* [24] and an extended version is actually under major revision for the *Data Mining and Knowledge Discovery* journal [30]. Chapter 5 details our second contribution about EMM with multi-label data. We explain a new EMM instance, that takes into account the high variance in the frequency of the class labels. Therefore, we define both a new model and a quality measure. We proceed to several batches of experiments to show the efficiency of this new EMM instance. This contribution has been published in the proceedings of the French conference *Extraction et Gestion de Connaissances EGC'15* [26] and the international conference *Discovery Science DS'16* [25]. Chapter 6 presents the results we obtained combining our two contributions in a real world application, namely the study the structure-odor relationship. We explain the SOR problem and its issues. We detail the data. We propose a study of the different rules computed with our algorithm. We implemented a Web application that enables the user to interact with the algorithm to guide the exploration. This application has been published as a demo paper in the European conference on *Machine Learning and Knowledge Discovery in Databases ECML/PKDD'16* [29]. Our collaborators are preparing a paper for the journal in neuroscience *PLOS One - Computational Biology*. Finally, Chapter 7 gives the conclusions and the perspectives of this thesis.

1.6 List of publications

- G. Bosc, J.-F. Boulicaut, C. Raïssi, M. Kaytoue:
  Découverte de sous-groupes avec les arbres de recherche de Monte Carlo.
  In Fabien Gandon and Gilles Bisson, editors, 17èmes Journées Francophones Extraction et Gestion des Connaissances, EGC 2017, Grenoble, France, 23-37 Janvier, 2017, volume E-33 of Revue des Nouvelles Technologies de l'Information, pages 273–284. Hermann-Editions, 2017 [24].

- V. Codocedo, G. Bosc, M. Kaytoue, J.-F. Boulicaut, A. Napoli:
  A Proposition for Sequence Mining Using Pattern Structures.
  In Karell Bertet, Daniel Borchmann, Peggy Cellier, and Sébastien Ferré, editors, Formal Concept Analysis - 14th International Conference, ICFCA 2017, Rennes, France, June 13-16, 2017, Proceedings, volume 10308 of Lecture Notes in Computer Science, pages 106–121. Springer, 2017 [47].

- G. Bosc, J. Golebiowski, M. Bensafi, C. Robardet, M. Plantevit, J.-F. Boulicaut, M. Kaytoue:
  Local Subgroup Discovery for Eliciting and Understanding New Structure-Odor Relationships.
  In Toon Calders, Michelangelo Ceci, and Donato Malerba, editors, Discovery Science - 19th International Conference, DS 2016, Bari, Italy, October 19-21, 2016, Proceedings, volume 9956 of Lecture Notes in Computer Science, pages 19–34. Springer, 2016 [25].

- G. Bosc, M. Plantevit, J.-F. Boulicaut, M. Bensafi, M. Kaytoue:
  h(odor): Interactive Discovery of Hypotheses on the Structure-Odor Relationship in Neuroscience.
  In Bettina Berendt, Björn Bringmann, Elisa Fromont, Gemma C. Garriga, Pauli Miettinen, Nikolaj Tatti, and Volker Tresp, editors, Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part III, volume 9853 of Lecture Notes in Computer Science, pages 17–21. Springer, 2016 [29].

- G. Bosc, M. Kaytoue, M. Plantevit, F. De Marchi, M. Bensafi, J.-F. Boulicaut:
  Vers la découverte de modèles exceptionnels locaux : des règles descriptives liant les molécules à leurs odeurs.
  In Benoît Otjacques, Jérôme Darmont, and Thomas Tamisier, editors, 15èmes Journées Francophones Extraction et Gestion des Connaissances, EGC 2015, 27-30 Janvier 2015, Luxembourg, volume E-28 of Revue des Nouvelles Technologies de l'Information, pages 305–316. Hermann-Editions, 2015 [26].

- G. Bosc, P. Tan, J. F. Boulicaut, C. Raissi, M. Kaytoue:
  A Pattern Mining Approach to Study Strategy Balance in RTS Games.
  IEEE Transactions on Computational Intelligence and AI in Games 9(2), pages 123-132 [31].

- G. Bosc, M. Kaytoue, C. Raïssi, J.-F. Boulicaut, P. Tan:
  Mining Balanced Sequential Patterns in RTS Games.
  In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014), volume 263 of Frontiers in Artificial Intelligence and Applications, pages 975–976. IOS Press, 2014 [28].

- G. Bosc, M. Kaytoue, C. Raïssi, J.-F. Boulicaut:
  Fouille de motifs séquentiels pour l'élicitation de stratégies à partir de traces d'interactions entre agents en compétition.
  In Chantal Reynaud, Arnaud Martin, and René Quiniou, editors, 14èmes Journées Francophones Extraction et Gestion des Connaissances, EGC 2014, Rennes, France, 28-32 Janvier, 2014, volume E-26 of Revue des Nouvelles Technologies de l'Information, pages 359–370. Hermann-Editions, 2014 [27].

Note that there is one paper in major revision:

- G. Bosc, J.-F. Boulicaut, C. Raïssi, M. Kaytoue:
  Anytime Discovery of a Diverse Set of Patterns with Monte Carlo Tree Search.
  Data Mining and Knowledge Discovery Journal, 48 pages. *Major revision, 2017 [30].*

Chapter 2

Pattern Mining

This chapter aims at introducing pattern mining and giving the main definitions we use in the rest of this document. First, we detail in Section 2.1 constraint-based pattern mining that has motivated a lot of works since the 90's. In Section 2.2, we define the problems of subgroup discovery (SD) and exceptional model mining (EMM), particular instances of pattern mining in which we are interested in this thesis. In Section 2.3, we discuss the different approaches proposed to solve them. Finally, we consider in Section 2.4 the several issues we finally address.

2.1 Constraint-based pattern mining

We formalize the constraint-based pattern mining framework and we introduce its most important research issues [33, 118].

Constraint-based pattern mining aims at proposing a generic framework to support the discovery of relevant patterns in data. The two main ideas are as follows. First, we must have the possibility to specify in a declarative way which are the patterns of interest in a given dataset: For that purpose, primitive constraints that are then combined thanks to boolean operators may be used to specify a so-called selection predicate that defines the patterns of interest among the collection of possible patterns, i.e., the sentences of a pattern language. Next, we must be able to compute the specified collection of patterns. This is where we have to discuss about algorithms and combinatorial complexity. This is where the possible enumeration strategies within the search space of the pattern language have to be considered seriously.

Basically, a pattern mining task or query[4] can be formalized as the theory of the data, following the terminology used in [111]:

$$Th(\mathcal{D}, \mathcal{L}, \Omega) = \{p \in \mathcal{L} \mid \Omega(p, \mathcal{D}) = true\}$$

$\mathcal{D}$ denotes the data, i.e., a set of objects (or instances), $\mathcal{L}$ denotes the pattern language, and Ω defines the selection predicate. Typically, such a predicate Ω is a boolean function that returns true if a pattern has to be kept in the result set, i.e., if the specified constraint that is a boolean combination of primitive constraints holds for the pattern. A quite common setting concerns the use of simple primitive constraints that look whether or not a quality measure is above or below a given threshold. One of the most popular example of such a primitive constraint concerns the minimal frequency constraint where the quality measure φ of a pattern $p \in \mathcal{L}$ denotes its

[4]Constraint-based mining is closely related to the inductive database vision where we try to define knowledge discovery processes as sequences of queries [32, 62]

number of occurrences in the data: To be frequent and thus to be considered interesting, a pattern frequency must be greater than the minimal frequency threshold. Indeed, such concepts have been studied for many types of data (transactions, sequences, collections of sequences, strings, trees, graphs, etc) and many types of pattern languages.

It is fairly easy to classify most of the existing works in the pattern mining area according to the simple concepts we have just introduced. Indeed, given a (type of) dataset $\mathcal{D}$, many authors contribute to the design or definition of new pattern languages (e.g., itemsets [4], sequences [6] or graphs [48]) associated to new primitive constraints that can eventually be based on new quality measures (see, e.g., [102]) to specify beforehand pattern relevancy. When the data, the pattern language and the primitive constraints are known, we can talk of a pattern domain.

Once we have a given pattern domain, the question of computing a collection of patterns that satisfy a given selection predicate can be extremely hard to solve. We have to enumerate the search space (sentences from $\mathcal{L}$) to find the correct and complete set of patterns, generally associated to the values of some of their quality measures. For instance, we want to compute every frequent pattern and its frequency. While many exhaustive algorithms have been designed for different pattern domains the last two decades, more and more interesting pattern mining tasks cannot be solved thanks to complete strategies. When it remains feasible, we can achieve scalable complete evaluation by exploiting the properties of the primitive constraints (e.g., the popular anti-monotonicity and monotonicity of many useful primitive constraints) to efficiently prune the search space during classical enumeration strategies (see, e.g., [84, 21, 44] when considering only a few results from our group in Lyon).

2.1.1 Mining frequent itemsets

A well-studied instance of pattern mining concerns the extraction of frequent patterns in transaction data. Frequent pattern mining aims at extracting substructures that appear in the data with a frequency greater or equal to a specified threshold. The frequent patterns give an overview over the data and are used to highlight the relationships among the data, e.g., while generating association rules [4]. Initially, frequent pattern mining deals with itemsets that have been studied a lot. Let $\mathcal{I}$ be a set of items. A transaction is a subset of items $t \subseteq \mathcal{I}$. The data $\mathcal{D}$ is a transaction database, that is a set of transactions $\mathcal{D} = \{t_1, ..., t_n\}$. An itemset is an arbitrary subset of items $p \subseteq \mathcal{I}$. The pattern language is the power-set of $\mathcal{I}$, i.e., the set of all itemsets: $\mathcal{L} = 2^{\mathcal{I}}$. We define a k-itemset an itemset of size k, i.e., containing k items.

Definition 1 (Extent and support of a pattern)**.** *The extent of a pattern $p \in \mathcal{L}$ in a transaction database $\mathcal{D}$ is the set of transactions of $\mathcal{D}$ that support the pattern p. Formally, the extent of p is:*

$$ext_{\mathcal{D}}(p) = \{t \in \mathcal{D} \mid p \subseteq t\}$$

Then, the support of p is the cardinality of its extent, that is the number of transactions that support p:

$$supp_{\mathcal{D}}(p) = |ext_{\mathcal{D}}(p)|$$

Note that, we omit the subscript mentioning the data in these definitions when there is no ambiguity about $\mathcal{D}$.

In frequent pattern mining, the quality measure $\varphi(p)$ of a pattern $p \in \mathcal{L}$ is its support $supp(p)$ and the selection predicate Ω is true if the support of a pattern p is greater or equal to a minimum support threshold, denoted as $minSupp$, specified by the user. Formally the selection predicate is $\Omega(p, \mathcal{D}) = supp(p) \geqslant minSupp$.

Table 6: A transaction database.

TID	items			
t_1	A	B		D
t_2	A		C	D
t_3		B		D
t_4		B	C	
t_5		B	C	
t_6	A	B	C	D

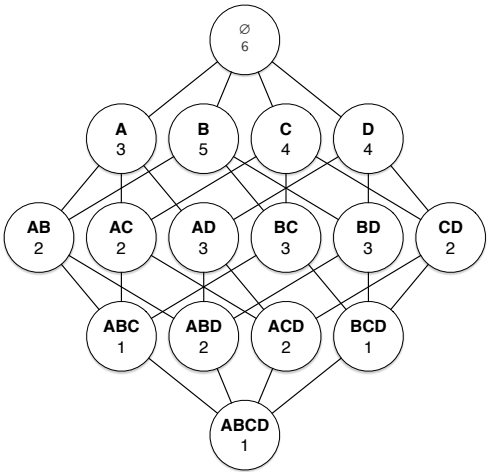


Figure 8: The lattice induced by the transaction database in Table 6. Each node is an itemset and the number is its support.

Problem 1 (Frequent itemset mining)**.** *Given a transaction database $\mathcal{D}$, a set of items $\mathcal{I}$ ($\mathcal{L} = 2^{\mathcal{I}}$) and a minimum support threshold $minSupp \in \mathbb{N}$, frequent itemset mining aims at performing the* correct *and* complete *extraction of all the frequent itemsets $p \in \mathcal{L}$ such that $supp(p) \geqslant minSupp$. Note that, in general, it goes further, i.e., both the frequent itemsets and their frequencies have to be extracted.*

Example 4. *Let us consider the toy transaction database $\mathcal{D}$ given in Table 6. This data contains 6 transactions identified by their TIDs $\{t_1, t_2, t_3, t_4, t_5, t_6\}$. We can suppose that each of these transactions is about the market basket of a customer. The set of items is $\mathcal{I} = \{A, B, C, D\}$. They can be the products bought by the customers. For example, the transaction t_2 contains the items A, C and D, meaning that the second customer has purchased the products A, C and D. The extent of the pattern $p = \{B, C\}$ is $ext(p) = \{t_4, t_5, t_6\}$ since $p \subseteq t_4$, $p \subseteq t_5$ and $p \subseteq t_6$: There are only three customers that have purchased the products B and C together. Thus, its support is $supp(p) = 3$. Given the minimum support threshold $minSupp = 3$, the set of frequent itemsets is $\{A\}$, $\{B\}$, $\{C\}$, $\{D\}$, $\{A, D\}$, $\{B, C\}$ and $\{B, D\}$.*

Computing frequent itemsets needs to explore the search space of itemsets. This search space is structured as a lattice $(2^{\mathcal{I}}, \subseteq)$. For instance, the lattice of the data in Table 6 is given in Figure 8. Each node of the lattice is an itemset, and the number at the bottom of each node is the support of the itemset. Since there is an exponential number of itemsets w.r.t. the number of items ($2^{|\mathcal{I}|}$ itemsets in the lattice), several algorithms have been proposed to efficiently enumerate

this search space[5]. Indeed, a naive exploration that enumerates each itemset is not scalable for large sets of items $\mathcal{I}$. We present two of the main mining methodologies that have been used to enumerate the itemsets, namely (i) the apriori principle and (ii) the FP-Growth strategy.

Apriori principle

The most famous property of the support is anti-monotonicity: If an itemset is frequent, then any of its sub-itemset is frequent [5]. Conversely, if an itemset is infrequent, any of its super-itemset is infrequent. Formally, let $p \in \mathcal{L}$ be an itemset and $minSupp$ be the minimum support threshold, if:

- $supp(p) \geqslant minSupp \Rightarrow \forall p' \in \mathcal{L}, p' \subseteq p, ext(p') \supseteq ext(p)$ and $supp(p') \geqslant minSupp$
- $supp(p) \leqslant minSupp \Rightarrow \forall p' \in \mathcal{L}, p' \supseteq p, ext(p') \subseteq ext(p)$ and $supp(p') \leqslant minSupp$

From this property, the Apriori algorithm is proposed by Agrawal and Srikant to extract the frequent itemsets in a top down approach. Indeed, the first step consists in finding the frequent 1-itemsets (itemsets of size 1). Then, the frequent 1-itemsets are used to generate the candidates for frequent 2-itemsets and the transaction database is used to check which of these candidates are frequent. The frequent 2-itemsets are used to find the frequent 3-itemsets and so on, until there is no more frequent k-itemsets.

Example 5. *Let us consider the transaction database given in Table 6 with* $minSupp = 3$. *The first step of the Apriori algorithm consists in finding the frequent 1-itemsets that are* $\{A\}$, $\{B\}$, $\{C\}$ *and* $\{D\}$. *From these frequent 1-itemsets, we generate the candidates for the frequent 2-itemsets:* $\{A, B\}$, $\{A, C\}$, $\{A, D\}$, $\{B, C\}$, $\{B, D\}$ *and* $\{C, D\}$. *For each of this candidate, we scan the database to check if they are frequent: Only* $\{A, D\}$, $\{B, C\}$ *and* $\{B, D\}$ *are frequent. Then, we generate the candidates for the frequent 3-itemsets: Only* $\{B, C, D\}$ *is generated but it is not frequent. The algorithm is stopped and the frequent patterns are* $\{A\}$, $\{B\}$, $\{C\}$, $\{D\}$, $\{A, D\}$, $\{B, C\}$ *and* $\{B, D\}$.

In many cases, the Apriori algorithm significantly prunes the search space. However, it can suffer from two-nontrivial costs: (i) generating a huge number of candidate sets, and (ii) repeatedly scanning the database and checking the candidates by pattern matching. About a decade later, another kind of exploration method has been proposed, namely the FP-Growth algorithm.

FP-Growth

FP-growth is an efficient and scalable method to find frequent patterns [80]. It allows to extract the frequent itemsets without candidate itemset generation (different from the Apriori algorithm). FP-growth employs a divide-and-conquer principle to extract the frequent patterns. Each recursion processes in two steps: (i) The construction of a data structure called the FP-Tree, and (ii) the extraction of frequent itemsets from the FP-Tree. The first scan of the transaction database derives the ordered list of the frequent 1-itemsets (decreasing order on the support of the frequent items). From this ordered list, the FP-Tree is built. Then, we mine the FP-tree by starting from each frequent 1-itemset, called the suffix, (from the less frequent to the most frequent), constructing its so-called conditional pattern database (CPD). From this conditional pattern base, we can derive its conditional FP-tree, and thus recursively mine this FP-Tree. We then concatenate the frequent patterns found in the FP-Tree with the suffix.

[5]Note that the frequent part of the lattice (i.e., patterns that are frequent) is known as the *iceberg* lattice [131].

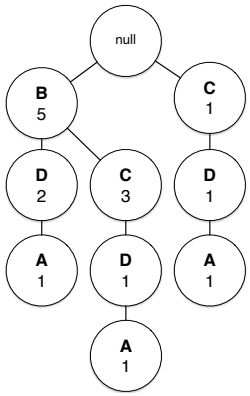


Figure 9: The FP-Tree of the transaction database in Table 6 with $minSupp = 3$.

Example 6. *From the transaction database of Table 6, we compute the ordered list of the frequent 1-itemsets:* $B[5]; C[4]; D[4]; A[3]$. *From that we can generate the FP-Tree (see Figure 9): For each transaction of the data, we ordered it w.r.t. the ordered list (e.g.,* t_1 *is* $\{B, D, A\}$*,* t_2 *is* $\{C, D, A\}$ *and so on) and we add the nodes and update the support count if a node already exists in the branch of the tree (e.g., for* t_1*, we add the node* $(B, 1)$ *as a child of the root node, then we add the node* $(D, 1)$ *as a child of* $(B, 1)$ *and finally the node* $(A, 1)$ *becomes a child of node* $(D, 1)$*; for* t_2 *we add the node* $(C, 1)$ *as child of the root node since the root node does not have the child* C *expanded yet, then we add* $(D, 1)$ *to* $(C, 1)$ *and* $(A, 1)$ *to* $(D, 1)$*; and so on). Based on this FP-Tree, for each frequent 1-itemset, we compute its conditional pattern base, iterating in the reverse order of the list. The CPD of* $\{A\}$ *is:*

- $B[1]; D[1]$
- $B[1]; C[1]; D[1]$
- $C[1]; D[1]$

Only $\{D\}$ *is frequent in the CPD of* $\{A\}$*:* $\{A, D\}$ *is added to the result set. The CPD of* $\{D\}$ *is:*

- $B[2]$
- $C[1]; B[1]$
- $C[1]$

Only $\{B\}$ *is frequent in the CPD of* $\{D\}$*:* $\{B, D\}$ *is added to the result set. The CPD of* $\{C\}$ *is:*

- $B[3]$

$\{B\}$ *is frequent in the CPD of* $\{C\}$*:* $\{B, C\}$ *is added to the result set. Finally the result set contains these frequent itemsets:* $\{A\}$*,* $\{B\}$*,* $\{C\}$*,* $\{D\}$*,* $\{A, D\}$*,* $\{B, C\}$ *and* $\{B, D\}$*.*

2.1.2 Beyond frequent itemset mining

Although frequent itemset mining enables to derive some knowledge from the data this pattern language lacks of expressiveness. Indeed, nowadays we are facing numerical health data that are collected from smart watches, social networks that are structured as graphs, or even sequences of images that highlight the movement of the celestial bodies in the universe. These data can

not be handled with itemsets: They require a more expressive pattern language. In addition, the pattern explosion is a well known property of frequent pattern mining approaches [39]. Depending on the minimum support threshold, there could be a huge amount of frequent patterns. This makes the interpretation of the results unfeasible. Moreover, the minimal frequency constraint is far from enough to specify interestingness and relevancy: Generally, the frequent patterns are not informative. To answer these different issues, several improvements of frequent itemset pattern mining are proposed in the state of the art, e.g., pattern compression with closed and maximal patterns to reduce the number of extracted patterns, pattern languages with a better expressiveness, or even novel quality measures to evaluate the interestingness of patterns.

Pattern compression

Usually, mining frequent itemsets in a transaction database leads to a huge number of frequent patterns [39]. This is particularly true when the minimum support threshold *minSupp* has to be low to avoid the extraction of trivial patterns. Due to the anti-monotonic property of the support, a large frequent pattern has an exponential number of frequent sub-patterns. To tackle this so-called pattern explosion, pattern compression allows to reduce the number of extracted patterns with or without a loss of information. For that, we define the equivalence class of a pattern $p \in \mathcal{L}$ [121]:

Definition 2 (Equivalence class)**.** *Given a transaction data $\mathcal{D}$, the equivalence class of an arbitrary itemset p is given by $[p] = \{p' \in \mathcal{L} | ext(p) = ext(p')\}$. In other words, the equivalence class of a pattern p is the set of patterns for which the extent is exactly the same.*

Example 7. *The equivalence class of the pattern $p_1 = \{A, C\}$ in the transaction database in Table 6 is $[p_1] = \{\{A, C\}, \{A, D\}, \{A, C, D\}\}$ since the extent of all of these itemsets is composed of t_2 and t_6.*

From the definition of the equivalence classes, closed patterns based enumeration has been proposed and has attracted a lot of works especially in the formal concept analysis (FCA) community [70]. Indeed, the closure operator enables to derive only one pattern from the equivalence class.

Definition 3 (Closed pattern)**.** *Each equivalence class has a unique largest element w.r.t. $\subseteq$ that is called the closed pattern (the most specific pattern). In other terms, p is said to be closed iff $\nexists p'$ such that $p \subseteq p'$ and $ext(p) = ext(p')$. The non-closed patterns are called generators, and the smallest generators w.r.t. the $\subseteq$ operator are called the minimal generators.*

The set of closed frequent patterns is a compression that does not lose any information about the corresponding frequent patterns. The main algorithms that aim at extracting frequent closed itemsets are CHARM [143], Close by One (CbO) [99] and its extensions Fast Close by One (FCbO) [97], AddIntent [136], and more recently the In-Close approach that is another variant of CbO [7, 8]. Most of them employ the closure operator with a lexicographical order on the items to ensure a canonical generation of the patterns.

Example 8. *In Table 6, let us consider the pattern $p_1 = \{A, C\}$ and its equivalence class $[p_1] = \{\{A, C\}, \{A, D\}, \{A, C, D\}\}$. The closed pattern is the largest itemsets w.r.t. $\subseteq$, i.e., $\{A, C, D\}$. Given the minimum support threshold $minSupp = 3$, the set of frequent closed patterns is composed of 6 itemsets: $\{B\}$, $\{C\}$, $\{D\}$, $\{A, D\}$, $\{B, C\}$, $\{B, D\}$. In this example, using frequent closed patterns enables to remove one pattern from the frequent pattern set, namely a compression of* 14%.

The other possibility to compress the frequent pattern set is to use the maximal frequent patterns [38, 75]. This solution enables to improve the compression rate but it leads to a loss of information: Indeed, it becomes impossible to recover the entire set of frequent patterns from the maximal frequent patterns. The algorithms that aim at extraction maximal frequent patterns are less numerous than for frequent closed patterns: The algorithms MAFIA [38] and GenMax [75] are two of the main ones.

Definition 4 (Maximal frequent pattern)**.** *Given a minimum support threshold minSupp, a pattern p is a frequent maximal pattern in a transaction database $\mathcal{D}$ if p is frequent and there is no super-patterns p' of p that is frequent. Formally, p is a frequent maximal pattern iff $\nexists p' \in \mathcal{L}$ such that $p \subseteq p'$ and $supp(p') \geqslant minSupp$.*

Example 9. *In the same settings than the previous example, the set of maximal frequent patterns contains only 3 itemsets, namely $\{A, D\}$, $\{B, C\}$, $\{B, D\}$. Indeed, the frequent 1-itemsets $\{A\}$, $\{B\}$, $\{C\}$ and $\{D\}$ are not maximal since there exists super-patterns that are frequent. Thus, the compression rate is 57%, four times better than those of the frequent closed patterns. However, once again, the maximal frequent pattern set does not enable to recover directly the entire set of frequent patterns with their frequencies, whereas the frequent closed pattern set does.*

Pattern language

Other improvements concern a growing expressiveness of the pattern language $\mathcal{L}$. Indeed, itemsets mining lacks of expressiveness in most of the real data. Nowadays, the collected data is not only binary. For example, the public opinion polls gather nominal (or categorical) value, the different sensors that are deployed all over the world return numerical values, the data collected in the supermarkets are made of sequences of itemsets, or the interactions in the social networks are represented with graphs. For that, in the previous decades there has been a lot of works that take into account this need for a growing expressiveness of the pattern language. In the following, we briefly present a pattern language for the nominal and numerical attributes. Note that sequential pattern mining has been introduced by Agrawal and Srikant [6][6]. Besides, graph mining has become an extremely active research domain that has steadily increased for a decade [48]. Enhancements of graph mining with attributed graphs [116] or dynamic graph [20, 124] have been studied. Recently, a work has proposed to mine convex polygons with formal concept analysis opening the road to geometrical shapes [16].

Nominal attributes are the generalization of itemsets where the domain of a nominal attribute is composed of several incomparable values. In the case of itemsets, the nominal attributes are the items, and each item can take the value true (present) or false (missing). Let $\mathcal{A} = \{A_1, A_2, \dots\}$ be the set of nominal attributes, the pattern language is the Cartesian product between the domain value $Dom(A_i)$ of each nominal attribute $A_i \in \mathcal{A}$: $\mathcal{L} = \bigtimes_{A_i \in \mathcal{A}} Dom(A_i)$.

Handling numerical attributes has motivated several works, in particular in FCA. A numerical pattern is the restriction on the numerical attributes using intervals. Thus let $\mathcal{A} = \{A_1, A_2, \dots\}$ be a set of numerical attributes, the pattern language is the set containing the Cartesian product

[6]We also worked on sequential pattern mining in the new context of electronic sports, in which an important challenge is to be able to detect game balance issues. For that, we presented an efficient pattern mining algorithm as a basic tool for game balance designers that enables one to search for imbalanced strategies in historical data through the KDD process. This work has been published, inter alia, in the international conference ECAI 2014 [28] and in the journal IEEE TCIAIG [31]. We do not detail this work in this dissertation. Interested readers may refer to the published papers available at `https://hal.archives-ouvertes.fr/hal-01252728/document` (IEEE TCIAIG) and `https://hal.archives-ouvertes.fr/hal-01100933/document` (ECAI 2014).

of all the possible intervals for each numerical attributes: $\mathcal{L} = \underset{A_i \in \mathcal{A}}{\times} [\alpha_i, \beta_i]$ where $\alpha_i, \beta_i \in Dom(A_i)$ and $Dom(A_i)$ is the domain value of the attribute A_i. The numerical patterns are also structured as a lattice [88, 89]. Figure 10 illustrates the upper part of the lattice on the numerical attributes of Table 7. Each child of a node consists in restricting the interval of an attribute either by a minimal left change, i.e., the lower bound is set to the next value (in ascending order) taken by this attribute in the data, or by a minimal right change, i.e., the upper bound of the interval is set to the previous value (in ascending order) taken by this attribute in the data. Still, the lattice is much larger than for itemsets, thus many works proceed to a discretization of the numerical attributes, either as a pre-processing task or an on-the-fly during the enumeration, to be equivalent to nominal attributes. However, even if the runtimes are faster, this naive strategy remains greedy and leads to sub-optimal results. In this thesis, we are interested in handling numerical data based on the minimal changes, to ensure the enumeration of all the possible intervals for each numerical attribute.

Quality measure

In addition to the pattern language, one major direction of research concerns the definition of new quality measures. Up to now, we simply present the support measure that is the initial quality measure that has been used in pattern mining. However, the support remains weak to evaluate the interestingness of a pattern. Indeed, the most frequent patterns are not necessary the more interesting because they can be trivial. That is why, scientists have worked and are still working about the definition of new quality measures to capture informative knowledge [102, 2, 96].

Mining frequent itemsets is used to discover association rules in a transaction database. Association rule mining was initially introduced by Agrawal et al. [4]. The aim is to find rules of the form $X \rightarrow Y$ where X and Y are disjoint itemsets. To evaluate an association rule, two measures are often used, namely the support and the confidence of the rule. An association rule is said interesting if its support is greater or equal to a given minimum support threshold $minSupp$ and its confidence is greater or equal to a given minimum confidence threshold $minConf$. The confidence of the rule $X \rightarrow Y$ measures the precision of the rule, i.e. the number of transactions containing X that also contains Y. It is defined as follows:

$$conf(X \rightarrow Y) = \frac{supp(X \cap Y)}{supp(X)}$$

In addition to the confidence, there are tens of measures that have been already defined to evaluate association rules. For instance, the lift measure of the rule $X \rightarrow Y$ quantifies the correlation between X and Y and is defined by:

$$lift(X \rightarrow Y) = \frac{supp(X \cap Y)}{supp(X) \times supp(Y)}$$

Constraint

In frequent itemset mining, the constraint consists in setting a minimum support threshold. Thus, a pattern is extracted if its support is greater than this threshold. This an anti-monotonic constraint. In the state of the art of constraint-based pattern mining, there was a trend to exhibit new properties of constraints and how to benefit from them: The relationship of constraint properties and enumeration principles is a fundamental issue. Anti-monotonicity is the first one that was studied with frequent pattern mining. However, there exists several other

anti-monotonic constraints: The maximum length of a pattern constraint, the maximum sum of cost constraint, and also the conjunction and the disjunction of anti-monotonic constraints. This property on constraints is not only available for itemsets, but also for other pattern languages, such as sequences or graphs. Anti-monotonic constraints enable to prune efficiently the search space when proceeding to a top-down enumeration: Indeed, if the constraint does not hold for a node, it does not hold for any of its children (a child is a specialization of its parent). Formally for a pattern language $\mathcal{L}$ with a partial order $\preceq$ (e.g., for itemsets $\preceq$ is $\supseteq$), $\forall p, p' \in \mathcal{L}, p' \preceq p, \Omega(p, \mathcal{D}) = \mathit{false} \Rightarrow \Omega(p', \mathcal{D}) = \mathit{false}$. This is the principle of the *Apriori* algorithm [5, 111]

Close to anti-monotonic constraints, monotonic constraints have been defined. This property can be seen as the negation of anti-monotonicity. If the monotonic constraint holds for a node in the lattice, it holds for all of its specialized patterns, i.e., its children. Formally, for a pattern language $\mathcal{L}$ with a partial order $\preceq$, $\forall p, p' \in \mathcal{L}, p' \preceq p, \Omega(p, \mathcal{D}) = \mathit{true} \Rightarrow \Omega(p', \mathcal{D}) = \mathit{true}$. The maximum support constraint is monotonic, as well as the minimum size constraint. Similarly to anti-monotonic constraints, with monotonic constraint, an efficient enumeration consists in exploring in a bottom-up approach, i.e., from the more specialized patterns to the more general ones, to efficiently prune the search space [111, 51].

Several properties have been identified to efficiently handle them during the enumeration by pruning the search space or computing upper bounds [115]. For example, convertible (anti-) monotonic constraints can be considered as (anti-) monotonic by using a specific enumeration: Thus, the previous pruning techniques can be applied. The aim of constraint-based mining is to design generic algorithms that exploit the different properties of the constraints and the pattern language [84, 21, 44]. However, sometimes it is difficult to state explicitly a constraint. In addition, setting a threshold is known to be hard because it would lead to a pattern explosion if the associated constraint becomes too loose or to the empty set of results if it turns to be too strong. For that, sky-patterns have been proposed: It aims at extracting the best patterns w.r.t. several quality measures by considering the Pareto frontier [130]. This enables to get rid of thresholds.

In summary, the trend in pattern mining is to efficiently extract patterns based on a given pattern language that is interesting w.r.t. quality measures. Association rules are the first attempt to exhibit some correlation in the data. However, recently, more and more datasets are about labeled data, i.e., transactions that are associated to a label. The supervised discovery of patterns that strongly distinguish one class label from another is still a challenging pattern mining task. Finally, pattern mining has become more and more able to extract interesting patterns leading to new knowledge. The model used to represent the data and the exploration methods are much more efficient to handle the existing relationships in the data. In this work, we are interested in the SD/EMM frameworks that enable to find exceptional behaviors within a labeled data.

2.2 Mining supervised descriptive rules

The discovery of patterns, or descriptions, which distinguish a group of objects given a target (class label) has been widely studied in data mining and machine learning [119]. Such descriptive rules can be formalized, among many other choices in AI, in subgroup discovery (SD, [93, 142]) or exceptional model mining (EMM, [104]). Two similar notions have been formalized independently and then unified by Novak et al. [119]: Contrast Set mining [15] and emerging patterns [56]. The first one aims to obtain high differences of support between the values of the target variable. The latter extracts patterns with different frequencies in two classes (e.g., the positive and the

negative classes) of the target variable. However, both methods and SD are similar and differ partially from the quality measure they use [119]. Close to SD, Redescription Mining aims to discover redescriptions of the same groups of objects according to different views [123, 137]. In this work we are interested in the SD/EMM frameworks. The input data is a population of individuals (objects, customers, transactions, ...) that embeds a set of descriptors and a set of class (or target) labels (some existing works deal also with numerical target attributes). We now detail both subgroup discovery (SD) and its generalization, the exceptional model mining framework (EMM).

2.2.1 Subgroup discovery

Subgroup discovery (SD) was first introduced in the middle of the 90's [93, 142]. The aim of SD is to extract subgroups of objects (described by a rule involving descriptors) for which the distribution on the target variable is statistically different from the whole (or, for some authors, the rest of the) population. Although these settings are related to those of a supervised learning task, SD is a descriptive task. A subgroup is a description generalization whose discriminating ability is assessed by a quality measure. Several measures have been proposed to evaluate a pattern w.r.t. a class label: The Weighted Relative Accuracy (WRAcc) [102], the F-Score, the Jaccard coefficients, or the weighted Kullback Leibler divergence (WKL). In the last two decades, different aspects of SD have been widely studied: The pattern language (quantitative, qualitative, etc.), the algorithms that enable the discovery of the best subgroups, and the definition of measures that express the interestingness or relevancy of a pattern. These three points are closely related, and many of the first approaches were *ad hoc* solutions lacking from easy implementable generalizations (see [59] for a survey).

Definition 5 (Label dataset)**.** *Let $\mathcal{O}$, $\mathcal{A}$ and $\mathcal{C}$ be respectively a set of objects, a set of attributes, and a set of class labels. The domain of an attribute $a \in \mathcal{A}$ is $Dom(a)$ where a is either nominal or numerical. Each object is associated to a class label from $\mathcal{C}$ through $class : \mathcal{O} \mapsto \mathcal{C}$. $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$ is a label dataset.*

Definition 6 (Subgroup)**.** *The description of a subgroup is given by $d = \langle f_1, \ldots, f_{|\mathcal{A}|} \rangle$ where each f_i is a restriction on the value domain of the attribute $a_i \in \mathcal{A}$. The description of a subgroup corresponds to the pattern in the pattern mining setting. A restriction is either a subset of a nominal attribute domain, or an interval contained in the domain of a numerical attribute. The description d covers a set of objects called the extent of the subgroup, denoted $ext(d) \subseteq \mathcal{O}$, and its support is the number of objects in its extent and is defined by $supp(d) = |ext(d)|$. Note that, in this thesis, we denote $|S|$ the cardinality of the set S. For simplicity, a subgroup is either given by its intent, i.e., its description d, or by its extent $ext(d)$.*

The ability of a subgroup to discriminate a class label is evaluated by means of a quality measure. The latter reflects the difference between the model induced by the subgroup on the target attribute and the model induced by the entire dataset. Basically, the model induced by a set of objects S is the proportion of objects of S associated to **one** class label $l \in \mathcal{C}$. The choice of the measure depends on the application [68]. There exists several quality measures that are used in SD, e.g., the WRAcc [102], Gini index, entropy or the Weighted Kullback-Leibler divergence (WKL) [2].

Problem 2 (The SD problem)**.** *Given a label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a quality measure φ, a minimum support threshold $minSupp$ and an integer k, SD aims at extracting the top-k best*

Table 7: Toy dataset

ID	a	b	c	$class(.)$
1	150	21	11	l_1
2	128	29	9	l_2
3	136	24	10	l_2
4	152	23	11	l_3
5	151	27	12	l_2
6	142	27	10	l_1

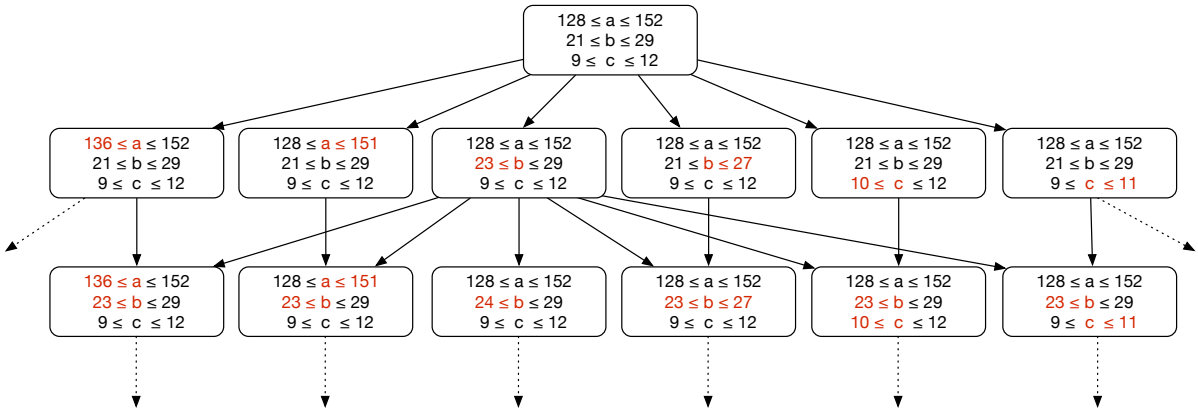


Figure 10: The upper part of the search space for the data of Table 7.

frequent subgroups w.r.t. φ. The quality measure φ quantifies a deviation between the model induced by $ext(d)$ and $\mathcal{O}$, respectively.

Note that, in the literature, the problem definition of SD can be slightly different from those we use in this thesis. Our definition is equivalent to those given in [59] in which we include a minimum support constraint in the set of constraints.

Example 10. *Consider the label dataset in Table 7 with objects $\mathcal{O} = \{1, ..., 6\}$ and attributes $\mathcal{A} = \{a, b, c\}$. Each object is labeled with a class label from $\mathcal{C}$. Considering the proportion class model, the proportion of the target label l_2 within the entire dataset is $p_0^{l_2} = \frac{|\{o \in \mathcal{O} | class(o) = l_2\}|}{|\mathcal{O}|} = \frac{1}{2}$. Consider the subgroup with description $d = \langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$. Note that for readability, we omit restrictions satisfied by all objects, e.g. $[9 \leqslant c \leqslant 12]$, and thus we denote that $ext(\langle\rangle) = \mathcal{O}$. The extent of d is composed of the objects $ext(d) = \{2, 3, 5, 6\}$. The distribution model for the target label l_2 in d is $p_d^{l_2} = \frac{|\{o \in ext(d) | class(o) = l_2\}|}{supp(d)} = \frac{3}{4}$. Then we compute if these distribution models are significantly different using the WRAcc measure for the label l_2: $WRAcc(d, l_2) = \frac{supp(d)}{|\mathcal{O}|} \times (p_d^{l_2} - p_0^{l_2}) = \frac{1}{6}$. The search space of the label dataset given in Table 7 is a lattice whose the top part is presented in Figure 10. We can notice the principle of minimal left and right changes to enumerate all the possible intervals.*

The SD problem has been extended in various ways, e.g., according to the pattern language, the algorithmic method to enumerate the search space, or more generally, the quality measure or exceptionality of a model induced by the subgroups. All these improvements can be discussed within the exceptional model mining framework (EMM [104]) that does not consider only one class label per object but several.

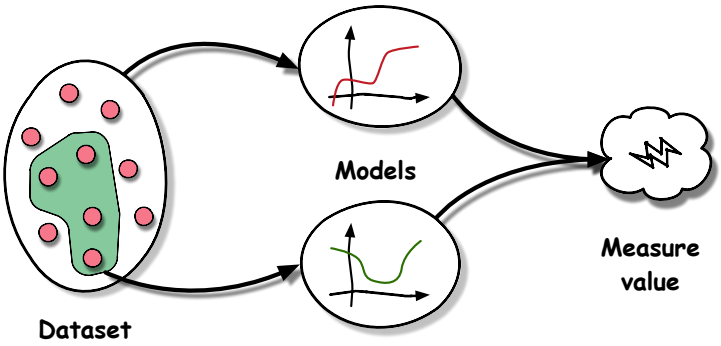


Figure 3: A typical EMM instance.

Table 8: The extension of the label dataset from Table 7 to the multi-label dataset version.

ID	a	b	c	$class(\cdot)$
1	150	21	11	$\{l_1,\ l_3\}$
2	128	29	9	$\{l_2\}$
3	136	24	10	$\{l_2,\ l_3\}$
4	152	23	11	$\{l_3\}$
5	151	27	12	$\{l_1,\ l_2\}$
6	142	27	10	$\{l_1,\ l_2\}$

2.2.2 Exceptional model mining

Exceptional model mining (EMM) was introduced by Leman et al. [104]. Recently, Duivesteijn et al. have proposed an interesting and complete survey on EMM [59]. It can be seen as a generalization of SD as it enables one to deal with more complex target concepts. Initially, SD aims at finding subgroups of objects for which the distribution over one value of the target variable deviates substantially from the distribution of the entire population of objects. In EMM, there is no longer one target variable but several ones: The descriptors are split into two sets, the descriptive variables and the target variables. The goal of EMM is to extract subgroups of objects for which the model induced over all the target variables substantially deviates from the model induced by the whole population of objects. Note that, in some works, the model induced by the subgroup has to deviate from the model of the complement set of objects and not the whole population. An illustration of this framework is given in Figure 3.

In this context, we have to override the definition of the label dataset given in Definition 5 as follows:

Definition 7 (Multi-label dataset)**.** *Let $\mathcal{O}$, $\mathcal{A}$ and $\mathcal{C}$ be respectively a set of objects, a set of attributes (either nominal or numerical), and a set of class labels. Each object is associated to a subset of class labels among $\mathcal{C}$ by the function $class : \mathcal{O} \mapsto 2^{\mathcal{C}}$ that maps the target labels to each object. We denote a multi-label dataset as $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$.*

In the EMM framework, given a multi-label dataset, the definition of a subgroup remains unchanged: Definition 6 is used. EMM relies on the model induced by a set of objects over the class labels. Many models have been proposed in the literature [59]: For instance, the probability distribution, a Bayesian Network, a clustering or a classification model over the class labels. Quality measure have been introduced to compare the similarity between the two models (i.e.,

those of the subgroup and those of the entire dataset), the better the quality measure, the less the similarity, the more interesting the subgroup. Figure 3 illustrates the process of EMM. On the left, the red circles are the objects of the dataset. The green area is the support of a subgroup. The red curve on the middle is the model induced by the entire dataset. The green curve is the model induced by the subgroup. These two models are compared by means of a quality measure. From that we can define the traditional EMM problem:

Problem 3 (The EMM problem)**.** *Given a multi-label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a quality measure φ comparing two instances of a class model, a minimum support threshold minSupp and an integer k, EMM aims at extracting the top-k best frequent subgroups w.r.t. φ. The quality measure φ quantifies a deviation between the model induced by $ext(d)$ and $\mathcal{O}$, respectively.*

Example 11. *Consider the multi-label dataset in Table 8 with objects $\mathcal{O} = \{1, 2, 3, 4, 5, 6\}$, attributes $\mathcal{A} = \{a, b, c\}$ and class labels $\mathcal{C} = \{l_1, l_2, l_3\}$. Each object is labeled with a subset of class labels from $\mathcal{C}$. Considering the probability distribution model class over the class labels, the aim is thus to find subgroups whose distribution over all the class labels is significantly different from those of the entire set of objects. The distribution of the entire dataset for each class label l_1, l_2 and l_3 is respectively $p_0^{l_1} = \frac{|\{o \in \mathcal{O} | l_1 \in class(o)\}|}{|\mathcal{O}|} = \frac{3}{6} = 0.5$, $p_0^{l_2} = 0.67$ and $p_0^{l_3} = 0.5$.*

Let us consider the subgroup s with description $d = \langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$. The extent of d is $ext(d) = \{2, 3, 5, 6\}$. The model induced by this subgroup for each class label is $p_d^{l_1} = \frac{|\{o \in supp(d) | l_1 \in class(o)\}|}{supp(d)} = \frac{2}{4} = 0.5$, $p_d^{l_2} = 1$ and $p_d^{l_3} = 0.25$. To compare these two models, we choose to use the mean of the WRAcc measures for each label:

$$\varphi(s) = \frac{4}{6} \times \left(\frac{(p_d^{l_1} - p_0^{l_1}) + (p_d^{l_2} - p_0^{l_2}) + (p_d^{l_3} - p_0^{l_3})}{3} \right) = 0.03$$

2.3 Algorithms for SD and EMM

SD and its generalization EMM have attracted a lot of works for more than two decades. Both exhaustive and heuristic methods have been used to explore the search space of subgroups.

2.3.1 Exhaustive exploration

The first exploration methods that have been proposed for SD implemented an exhaustive search, i.e., the search space is completely explored ensuring that the best subgroups are found [87, 12, 11]. Several pruning strategies are used to avoid the exploration of uninteresting parts of the search space. These pruning strategies are usually based on the monotonic (or anti-monotonic) property of the support or the quality measures.

The first algorithm *EXPLORA* [93] is an exhaustive approach that employs decision trees. The *MIDOS* algorithm [142] is also based on decision trees with minimum support pruning. These methods are extensions of classification algorithms. The pattern language used to generate the rules is made of conjunctions of pair attribute-value. The target variable is either nominal or binary. Another kind of algorithms extends association rule learner algorithms such as the *APRIORI-SD* algorithm [87]. It also employs conjunctions of pairs attribute-value with a nominal target variable.

Some exhaustive approaches employ efficient pruning methods to explore the search space. *Merge-SD* performs large pruning of the search based on bounds [76]. This algorithm is able to deal with numerical attributes. The *BSD* algorithm uses a bitset (bitvector) based data structure with a depth-first-search approach [107]. Recently, another approach has been proposed to

efficiently perform an exhaustive exploration with a numerical target variable with optimistic estimates on different quality measures [105]. The authors also compared different data structures, e.g., FP-trees or bitset.

Efficient methods for SD deal with the use of efficient data structures, often the well-known FP-growth method [80] and a bitset structure. *SD-MAP* [12] and *DpSubgroup* are based on FP-trees with a binary and nominal target variable. *SD-MAP* exploits a minimum support threshold whereas *DpSubgroup* [77] employs tight optimistic estimates. *SD-MAP** [11] extends the *SD-MAP* algorithm to numerical target variables.

To the best of our knowledge, there is only one exhaustive algorithm, namely *GP-Growth*, that addresses explicitly EMM [106]. It extends the previous works done with *SD-MAP* and *SD-MAP** for traditional SD. *GP-Growth* is based on the *FP-Growth* algorithm that uses the efficiency of the FP-Tree structure. *GP-Growth* employs a valuation basis that enables to get rid of the previous structure in which the frequency is included in each node in the tree. Thus, the kind of information stored in the valuation basis depends on the model class. Finally, the authors adapted the *FP-Growth* algorithm where the tree structure called the GP-Tree no longer stores frequencies but valuation bases. So, the algorithm *GP-Growth* proceeds to the aggregation of valuation bases for each pattern. Note that, this algorithm, as well as *SD-MAP* and *SD-MAP**, uses an equal width discretization to handle numerical attributes. This pre-processing task avoid the exploration of the whole dataset by turning numerical attributes to nominal attributes, and it leads to an non-optimal result set.

2.3.2 Heuristic search

Exhaustive search ensures to extract the best subgroups, however its main drawback is that it is not scalable when the data becomes large. Such data leads to enormous search space making exhaustive search infeasible. Indeed, the more the descriptors, the longer the runtime. Similarly, the more the individuals, the longer the runtime (excepted for FP-growth methods that do not enumerate the individuals). To tackle the runtime issue, heuristic approaches have been used, ensuring a shorter exploration, but there is no guaranties to extract the best subgroups. For that, several heuristics have been used.

Beam search strategy

First, the commonly used search strategy is based on the beam search [108]. This heuristic enables to perform a level wise exploration of the search space, avoiding the so-called uninteresting parts, with several hill climbings run in parallel. A beam of a given size (or dynamic size for recent works) is built from the root of the search space. This beam only keeps the best promising subgroups to extend w.r.t. the quality measure φ. The first algorithms for SD designed with a beam search strategies are *SubgroupMiner* [94], *SD* [69] and *CN2-SD* [101]. Then, the algorithm *SD4TS* is developed [117], based on *APRIORI-SD*. It uses the quality measure of the subgroups to prune the search space.

More recently, several works have been done about beam search for EMM. One of the first heuristic algorithms is developed by van Leeuwen et al. in [138]. The main point of this work is about redundancy in "Generalized Subgroup Discovery" (see the following Subsection 2.4 for more information about subgroup set discovery). Moreover, the authors present several beam strategies and compare them experimentally. Then, they extended this work [139]. In this paper, the heuristic algorithm *DSSD*, Diverse Subgroup Set Discovery, is presented. This algorithm employs a beam search depending on several beam strategies. The experimental study includes a

much wider range of datasets with deep comparisons. *DSSD* acts as a baseline algorithm for both SD and EMM. The following year, the algorithm *DSSD* is extended with a skyline approach that deals with the issue of redundancy in the top-k approaches [140]. We will discuss this redundancy issue in Subsection 2.4 as well.

ROCsearch is a new ROC-based beam search variant for EMM [112]. Following the trend inspired by van Leeuwen et al., the authors have implemented a new selection method for beam search strategies. Indeed, commonly used beam search employs selection strategies that are only based on the quality measure. Some, as in the work of van Leeuwen et al. [138], try to include others parameters that are able to avoid redundancy. However, these parameters often require thresholds that have to be fixed by the user. Fixing a parameter is well-known to be hard. *ROCsearch* proposes a new beam search strategy that enables to avoid fixing parameters. The width of the beam is dynamically set, depending on the current state of the previous beam. The selection strategy is based on the ROC space of the subgroups. The authors claim that the diversity of the result set is improved and the redundancy is decreased.

Evolutionary algorithms

The second mainstream of heuristic approaches for SD is about evolutionary algorithms [14]. These approaches aim at solving problems imitating the process of natural evolution. Genetic algorithms are a branch of the evolutionary approaches that use a fitness function to select which individuals to keep at the next generated population [83]. *SDIGA* [54] is one of the first evolutionary algorithms designed for *Subgroup Discovery*. It is based on a fuzzy rule induction system. A fuzzy rule corresponds to the description of a subgroup, and is written in disjunctive normal form (DNF). Several quality measures are embedded in *SDIGA* such as support, confidence or sensibility. The objective function is an aggregating function based on several quality measures such as the coverage, the significance, the support or the confidence. *EDER-SD* [125] is a sequential covering evolutionary algorithm that produces a hierarchical set of rules. This algorithm extracts subgroups on imbalanced data, and especially the subgroups which target variable is the minority class of the dataset. Similarly to *CN2-SD*, *EDER-SD* penalizes the objects that are already covered by other rules (or subgroups). It can be used for numerical target variables. The objective function is based on different quality measures usually used in SD: accuracy, F-Score, sensitivity, ... *GAR-SD* [120] is an evolutionary multi-objective algorithm. It can work with both discrete and continuous attributes without previous discretization. It is based on the Iterative Rule-Learning approach. The objective function used in *GAR-SD* uses three different quality measures derived from SD (support, confidence and significance) and three other measures based on the characteristics of the rule. *GP3-SD* [110] is based on a genetic programming algorithm for mining association rules for each value of the target variable. The descriptions of the subgroups are represented thanks to tree structures. Contrary to the previous algorithms, *GP3-SD* does not employ an Iterative Rule-Learning approach but the elitism for the selection procedure. The objective function is based on the support and the confidence of the rules. *MESDIF* [19] is a multi-objective genetic algorithm that extracts rules describing subgroups. This algorithm can generate fuzzy and/or crisp DNF rules, for problems with continuous and/or nominal variables. It searches for optimal solution in the Pareto front based on the *SPEA2* approach [145]. *NMEEF-SD* [40] is a multi-objective evolutionary fuzzy algorithm. It is based on *NSGA-II* [53], an efficient multi-objective evolutionary algorithm, and on the use of elitism. It allows the user to choose several quality measures as objective functions such as the support, the confidence or even the accuracy.

Sampling methods

Another trend of heuristic search for EMM is about sampling methods. A recent work employs Controlled Direct Pattern Sampling (CDPS) to search for subgroups in the EMM framework [114]. CDPS is a sampling method that enables to create random patterns thanks to a procedure based on a controlled distribution [23]. The authors adapt this sampling methods to EMM to foster subgroups for which the model deviates from those which hold for the whole dataset and with high frequency. More recently, a work employs a sampling method to discover exceptional models induced by attributed graphs [17]. In this work, the authors create a distribution to give more chance to an interesting subgraph to be drawn. The model is based on the characteristics induced by the subgraph. Pattern sampling is attractive as it allows direct interactions with the user for using his/her preferences for driving the search: A result is available anytime. However, traditional sampling methods used in pattern mining need a given probability distribution over the pattern space: This distribution depends on both the data and the measure [23, 114]. This probability distribution has to be computed in a pre-processing step. Each iteration is independent and consists of drawing a pattern given this probability distribution. Moreover, these probability distributions exhibit the problem of the long tail: There are many more uninteresting pattern than interesting ones. Thus, the probability to draw an uninteresting pattern is still high, and not all local optima may be drawn: There are no guaranties on the completeness of the result set.

2.4 The main issues about descriptive rules mining

Subgroup discovery and its generalization exceptional model mining highlight several issues that have to be addressed to propose correct and efficient approaches for supervised descriptive rule discovery. The existing works focus on the definition of new models with their quality measures to capture more interesting patterns. The issue of diversity in the result set is at the core of heuristic searches: How to ensure a good diversity in the results when applying heuristic exploration methods? In addition, how interactive algorithms can improve the extraction of more interesting subgroups by taking into account the expert's preferences during the exploration? Finally, how to ensure an efficient extraction of the patterns by exhaustive or heuristic searches? In the following, we detail and discuss the existing approaches to handle these issues.

Choosing the appropriate model

EMM is based on a class model, and the aim is to search for subgroups of individuals for which the model induced on the target attributes deviates significantly from the model of the whole dataset (or of the complement of the subgroup). Several models have been proposed in the state of the art [59], e.g., the linear regression between two numerical class attributes [58], or the contingency tables for several class labels [114] or the Bayesian networks computed on the class labels [61]. The choice of the model depends on both the target attributes and the purpose of the application [58]. The main stream of works about EMM consists of finding new class models to deal with specific datasets and objectives. In [59], the authors define lots of possible models that have been used. The simplest models deal with the correlation (or association) between two target attributes: The aim is to find subgroups for which the correlation is significantly different from those induced on the entire dataset. One can also be interested in the difference of the linear regression of one numerical target attribute in the subgroups, by comparing for instance the slope of these models. More complex models can also be suitable for specific cases, e.g.,

the models based on Bayesian networks. In this setting, it is assumed that there are multiple nominal target attributes. A subgroup is considered as interesting if the conditional dependency relations between the target attributes are significantly different in the subgroup from those of the entire dataset. For that, the Bayesian networks of both the subgroups and the entire dataset are compared [61]. Recently, a new model has been proposed to handle ranking data. The Rank Correlation Model Class has been introduced to be less sensitive to the outliers that are in the target attributes [57]. This class model no longer uses the values taken by the target attributes but their rank. This method only works if there are two target attributes that can be ordered (to get the rank of the values): Ordinal or numerical attributes are taken into account.

In our work, since the experts of the domain are interested in descriptive rules that conclude on few labels, we propose a new EMM instance that considers several target subspaces. In other words, from a subgroup, several models are derived: A model is built on each subset of labels. The model we define is based on both the precision and the recall of the subgroup w.r.t. a given target subspace.

Quality measures

Once the model is chosen, it is required to select the quality measure that compares two instances of the models. The quality measure is the core of the method in SD or EMM. Indeed, it enables to quantify the difference between the model induced by the subgroup and those of the entire dataset. The quality measure depends on both the model and the purpose of the application. There exists a large pool of quality measures in the literature [102, 2]. Usually, when the application is specific, it requires to design a new quality measure that enables to encode the needs of the experts, i.e., the so-called subjective interestingness. Konijn et al. present several quality measures when facing datasets with two target attributes: A binary attribute that corresponds to the main class with the values positive or negative and the second target attribute represents a cost [96]. From that, they designed several cost-based quality measures that enable to find out interesting subgroups for which the cost target attribute differs for positive examples from the cost attribute values of the whole dataset. Moreover, they employ a new exploration method based on the Local Subgroup Discovery (LSD). Contrary to the traditional Descriptive Subgroup Discovery (DSD) that performs a top down exploration by extending the description, LSD enables to zoom in a part of the dataset: It searches for the nearest neighbors of a reference subgroup to detect close subgroups that deviate from this reference group. Thus, the aim is to find out a subgroup that is close to a reference subgroup but for which the cost attribute is really different from this reference subgroup. Duivesteijn et al. employ a new model class for EMM [61]. They propose to use the interdependencies of the target variables to quantify the quality of the subgroups. For that, they associate EMM with a model class based on Bayesian networks on the target variables. They design several quality measures to evaluate the distance between two Bayesian networks. In the case of regression models, Duivesteijn et al. proposed to use the Cook's Distance [58]. The Cook's distance does not require to normalize the attributes. There are also some theoretical upper bounds on this distance that enables to improve the efficiency of this method by pruning. Concerning the Rank Correlation Model Class, Duivesteijn et al. present several correlation measures based on the ranks of the target attributes that enable to compute the exceptional nature of the subgroups (Spearman's Rank Correlation coefficient, Kendall's Tau, ...). Recently, exceptional preferences mining has been defined [52] to extract subgroups where the preference relations between subsets (using label ranking) of class labels significantly deviate from those of the entire dataset. Finally, the extracted subgroups have to bring new information to the experts: We are not interested in obvious subgroups. For that, most of

the existing works implement a statistically significance test. For that, they test the distribution of a subgroup against the null hypothesis. If the null hypothesis is rejected, it means that the subgroup is significantly informative.

In our work, we propose several quality measures to compare two instances of the model class we introduce. For that, since the distribution of the labels is skewed, we take into account this high variance in the frequency of the labels to compute the quality measure.

Redundancy and diversity

The redundancy is one of the main issues for SD or EMM to ensure enough diversity in the result set. Indeed, once the subgroups have been extracted, the result set contains many duplicates. Indeed, there are several subgroups that are related to the same local optimum: Many variants of the same pattern exist. For example a slightly different interval for a numerical attributes induces a little change in the subgroup of objects. This causes top-k mining algorithms to return highly redundant result sets, while ignoring many potentially interesting results. To avoid the redundancy in the result set and to improve the diversity, many researchers proposed to filter out redundant subgroups. A first attempt was proposed by van Leeuwen et al. [138]. This work not only considers the subgroups as individual ones but as a set of subgroups. For that they introduced the *subgroup set mining* task. They proposed a similarity measure to filter out redundant subgroups based on the subgroup descriptions and the extents of the subgroups. They also used this similarity measure to implement several beam selections to take into account the redundancy at each level of the beam search. They introduced the Diverse Subgroup Set Discovery algorithm that implements these methods [139]. To go further, van Leeuwen et al. also proposed to use skylines methods to deal with redundant subgroups [140]. In fact, considering the Pareto front ensures to avoid some of the redundant subgroups of the result set.

In our work, to remove redundancy in the result set, we apply the support-based similarity measure that is used in most of the existing works in SD/EMM. This similarity measure is applied in a post-processing step, to filter out redundant subgroups from the result set.

Algorithmic improvements

Many papers address algorithmic issues for SD and EMM. Considering the exhaustive searches, the major improvement is about the data structure used. The paper of Atzmüller and Puppe [12] proposes to use the well known *FP-Growth* approach to deal with subgroup discovery. It results in great improvements of the efficiency of the extraction in terms of runtime. Moreover, the use of efficient pruning strategies is an alternative to reduce the runtime of the exploration. In [77], the authors propose to implement tight optimistic estimates to prune uninteresting parts of the search space. They are able to compute an upper bound on the quality measure of a subtree, and if it is too low, they prune this subtree since it will not provide better results. Finally, several algorithms implement the parallelization of the execution of the source code. In this way, the runtime reduces. This parallelization is more or less efficient w.r.t. the exploration method of the search space: Some exploration methods are more likely to be processed in a parallelization way than others. However, most of the recent works have focused on beam search and do not discuss the choice of this greedy method that really poorly explores the search space. In fact, they are able to extract only few local optima from the search space (see Figure 6b)).

Later, we define the *completeness* issue as the ability of an heuristic method to extract as many local optima as possible. In our work, we propose a new enumeration method based on

Monte Carlo tree search to address the completeness issue [36].

Instant mining and interactivity

A new trend of SD/EMM methods is related to the instant mining approaches. Basically, SD or EMM are frameworks that extract interesting subgroups that are easily understandable and actionable by the experts. The end user, i.e., the expert, is on the core of SD/EMM. The description of the subgroup is constructed to be as simple as possible. In this matter, instant mining methods provide a great advantage to an end user application. For that, sampling approaches are efficient as they provide anytime a solution [114]. The longer the time budget, the better the result set. To a lesser extent, beam search and exhaustive search are less able to be considered as instant mining methods. In fact, their greedy approach fails at providing interesting patterns anytime. To go further with instant mining tools, many recent algorithms are designed to interact with the experts. In this way, the expert can guide the exploration to foster on the areas within the search space he/she prefers. Typically, with a beam search, the expert can add or remove some subgroups from the beam. Thus, the exploration of the next level is guided by the expert's preferences [63].

In our work in olfaction, we implement an interactive Web application to take into account the expert's preferences to guide the exploration. This application is used by the neuroscientists and the chemists to elicit new hypotheses on the structure-odor relationship.

2.5 Conclusion

Subgroup discovery has attracted a lot of attention since its introduction in the middle of the 90's. Its generalization into the exceptional model mining framework enables to take into account more complex hidden relations in the data. The definition of new models with specific quality measures has widened the scope of applications. Although several issues have been solved (such as the redundancy problem in the result), it remains some open problems that need to be addressed: For example, the development of a new anytime heuristic method that enables to efficiently explore the search space without any knowledge on the distribution of the quality measure over it. Indeed, most of the recent works employ a beam search strategy that greedily explores the search space with several hill climbings run in parallel. Clearly, it can not result in a set of subgroups that covers most of the local optima. As our major contribution, we propose to tackle this issue by developing a new heuristic algorithm for SD/EMM based on Monte Carlo tree search.

Chapter 3

Monte Carlo Tree Search

Monte Carlo tree search (MCTS) is a search method used in several domains for finding an optimal decision. It relies on taking successive random samples of the search space and build or update a tree accordingly [36]. MCTS merges different theoretical results obtained from decision theory, game theory, Monte Carlo and bandit-based methods. We introduce preliminary notions and definitions before developing the basic MCTS algorithm, called UCT. There are lots of works that deal with MCTS. Here, we present only what is needed to understand our contribution in the next chapter. An exhaustive survey about MCTS until 2012 is [36].

3.1 Background

3.1.1 Decision theory

Studying which decision to take w.r.t. a set of observations and past actions is one of the most explored problem in the 20^{th} century [126]. Several models have been implemented to represent and understand such a phenomena. For instance, the Markov Decision Problem (MDP) is applied to model sequential decision problems. This model relies on a set of states S, a set of actions A, a transition model $T(s, a, s')$ that determines the probability to reach the state s' from the state s playing the action a, and a reward function $R(s)$ that evaluates the quality of reaching a state s. Therefore, the decisions are modeled with sequences of state-action pairs (s, a). The aim is to find policies that specify which action to play from each state. The higher the reward the better the policy.

3.1.2 Game theory

Game theory is the extension of the decision theory in which several agents can interact during the process. Considering a set of states S containing terminal states $S_T \subseteq S$, the reward function usually assigns to a terminal state a reward among $\{-1, 0, 1\}$ for a loss, draw or win, and the reward 0 for a non terminal state. Each player's strategy evaluates the probability for a player to choose an action given a specific state. In some cases, the players' strategies can form a so called Nash equilibrium, if none of the players can obtain a better reward by unilaterally switching his/her strategy [126]. The games are described by several properties w.r.t. their characteristics: The zero-sum games (the sum of all the players' rewards is null), the information aspects of the games (complete or not), the determinism (the game contains random transitions or not), sequential games (the actions are chosen sequentially or simultaneously), real time or discrete

games, etc. To explore these games, one of the most famous strategy is Minimax that tries to minimize the reward of the opponents while maximizing its own reward.

3.1.3 Monte Carlo methods

Determining which action to play given a specific state can also be computed thanks to sampling methods such as Monte Carlo methods [1]. This approach is used to approximate the game-theoretic value of an action by performing several simulations of the game from the current state to a terminal state. Then, the player can select the action that experimentally leads to better rewards. However, this naive sampling method does not use the past experiences of choosing an action to bias the choice of the following simulations. For that, the theoretical results obtained with bandit-based methods are useful.

3.1.4 Bandit based methods

The bandit problem is a well known class of sequential decision problems. Considering a multi-armed bandit slot machine containing K arms, the goal is to choose the arm that maximizes the cumulative rewards. Each arm of the machine follows a probability distribution of its reward that is unknown by the player. The aim is to find the arm that proposes the best expected reward by playing as few times as possible. However, the potential rewards can be estimated thanks to the past observations. This leads to an exploration/exploitation trade-off: The player wants to play with the arm that gave the best rewards so far (exploitation), but he has also to play with the other arms that may turn out to be superior in the long run (exploration). For that, the player's regret is considered. After N plays the player's regret is:

$$R = \mu^* N - \sum_{j=1}^{K} \mu_j \mathbb{E}[T_j(N)]$$

where $\mu^* = \max_{j=1}^{K} \mu_j$ is the best possible expected rewards, μ_j is the expected reward of the arm j and $\mathbb{E}[T_j(N)]$ is the expected number of times the arm j is played in the first N trials. Thus, the regret is the expected loss due to not playing the optimal arm. Determining the optimal arm to play can be done thanks to confidence bounds. For example, the upper confidence bound policy proposed in [13], called UCB1, has an expected logarithmic growth of regret uniformly over N. This policy assumes that the player has to play the arm A_j that maximizes:

$$UCB1(A_j) = Q(A_j) + \sqrt{\frac{2 \ln N}{N(A_j)}}$$

where $Q(A_j)$ is the average reward of the arm A_j obtained so far, $N(A_j)$ is the number of times the arm A_j has been played and N is the overall number of plays. The term $Q(A_j)$ encourages the exploitation of interesting rewards, whereas the term $\sqrt{\frac{2 \ln N}{N(A_j)}}$ encourages the exploration of few played actions.

3.2 The UCT algorithm

MCTS is a search method used in several domains to find an optimal decision (see [36] for a survey). It merges theoretical results from decision theory [126], game theory, Monte Carlo [1]

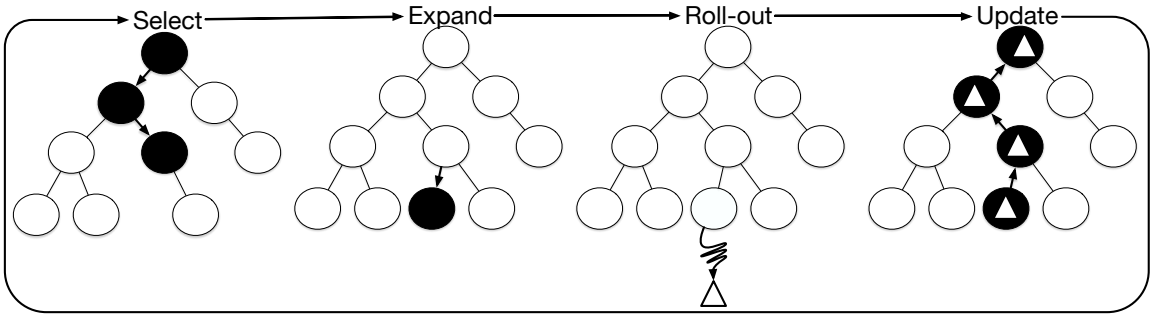


Figure 11: One MCTS iteration (inspired from [36]).

and bandit-based methods [13]. MCTS is a powerful method because it allows the joint use of random simulations and a trade-off between exploration of the search tree and the exploitation of an interesting solution based on past observations [36]. Considering a game, e.g., Go, the goal of MCTS is to find the best action to play given a current game state. MCTS proceeds in several (limited) iterations that build a partial game tree (called the search tree) depending on the results of the previous iterations. The nodes represent the game states. The root node is the current game state. The children of a node are the game states accessible from this node by playing an available action. The terminal nodes are the terminal game states. Each iteration, consisting of 4 steps (see Figure 11), leads to the generation of a new node in the search tree (depending on the exploration/exploitation trade-off due to the past iterations) followed by a simulation (sequence of actions up to a terminal node). Any node s in the search tree is provided with two values: The number $N(s)$ of times it has been visited, and a value $Q(s)$ that corresponds to the aggregation of rewards of all simulations walked through s so far (e.g., the proportion of wins obtained for all simulations walked through s). The aggregated reward of each node is updated through the iterations and becomes more and more accurate. Once the computation budget is reached, MCTS returns the best move that leads to the child of the root node with the best aggregated reward $Q(.)$. In the following, we detail the 4 steps of a MCTS iteration applied to a game. Algorithm 1 gives the pseudo code of the most popular algorithm in the MCTS family, namely UCT (Upper Confidence bound for Trees) [95].

The Select policy

Starting from the root node, the Select method recursively selects an action (an edge) until the selected node is either a terminal game state or is not fully expanded (there remain children of this node that are not yet expanded in the search tree). The selection of a child of a node s is based on the exploration/exploitation trade-off. For that, upper confidence bounds (UCB) are used. They estimate the regret of choosing a non-optimal child. The original UCBs used in MCTS are the UCB1 [13] and one of its variant, namely the UCT:

$$UCT(s, s') = Q(s') + 2C_p\sqrt{\frac{2\ln N(s)}{N(s')}}$$

where s' is a child of a node s and $C_p > 0$ is a constant (generally, $C_p = \frac{1}{\sqrt{2}}$). This step selects the most urgent node to be expanded, called s_{sel} in the following, considering both the exploitation of interesting actions (given by the first term in UCT) and the exploration of lightly explored areas of the search space (e.g., given by the second in UCT) based on the result of past iterations.

Algorithm 1 UCT: The popular MCTS algorithm.

function MCTS($budget$)
    create root node s_0 for current state
    while within computational budget $budget$ **do**
        $s_{sel} \leftarrow$ SELECT(s_0)
        $s_{exp} \leftarrow$ EXPAND(s_{sel})
        $\Delta \leftarrow$ ROLLOUT(s_{exp})
        UPDATE(s_{exp}, Δ)
    end while
    return the action that reaches the child s of s_0 with the highest $Q(s)$
end function

function SELECT(s)
    while s is non-terminal **do**
        if s is not fully expanded **then** **return** s
        else $s \leftarrow$ BESTCHILD(s)
        end if
    end while
    return s
end function

function EXPAND(s_{sel})
    randomly choose s_{exp} from non expanded children of s_{sel}
    add new child s_{exp} to s_{sel}
    return s_{exp}
end function

function ROLLOUT(s)
    $\Delta \leftarrow 0$
    while s is non-terminal **do**
        choose randomly a child s' of s
        $s \leftarrow s'$
    end while
    return the reward of the terminal state s
end function

function UPDATE(s, Δ)
    while s is not null **do**
        $Q(s) \leftarrow \frac{N(s) \times Q(s) + \Delta}{N(s)+1}$
        $N(s) \leftarrow N(s) + 1$
        $s \leftarrow$ parent of s
    end while
end function

function BESTCHILD(s)
    return $\underset{s' \in \textit{ children of } s}{\arg\max} \; UCB(s, s')$
end function

The constant C_p can be adjusted to lower or increase the weigh of exploration in the trade-off exploration/exploitation. Note that when $C_p = \frac{1}{2}$, this is UCB1.

The Expand policy

A new child, denoted s_{exp}, of the selected node s_{sel} is added to the tree according to the available actions. The child s_{exp} is randomly picked among all available children of s_{sel} not yet expanded in the search tree.

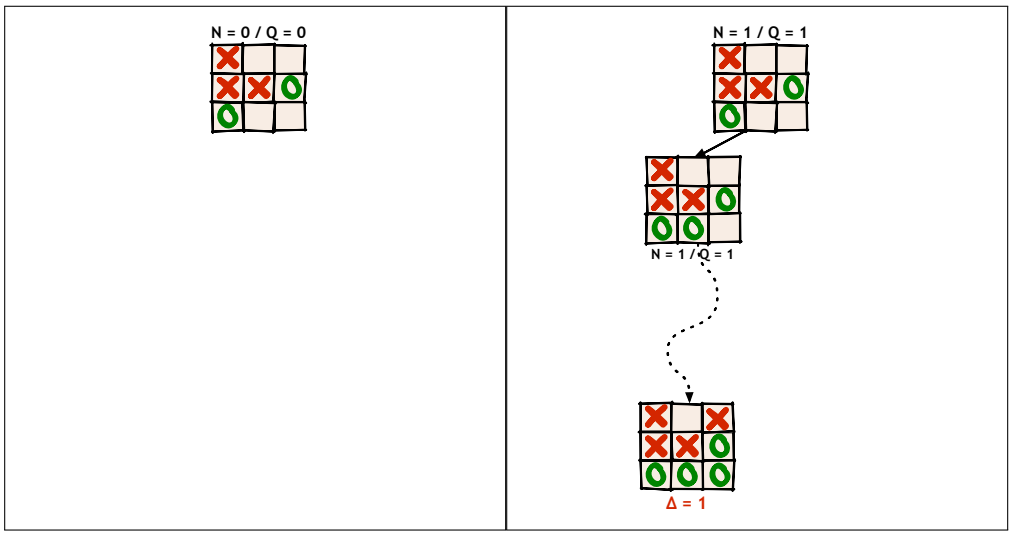
The RollOut policy

From this expanded node s_{exp}, a simulation is played based on a specific policy. This simulation consists of exploring the search tree (playing a sequence of actions) from s_{exp} until a terminal state is reached. It returns the reward Δ of this terminal state: $\Delta = 1$ if the terminal state is a win, $\Delta = 0$ otherwise.

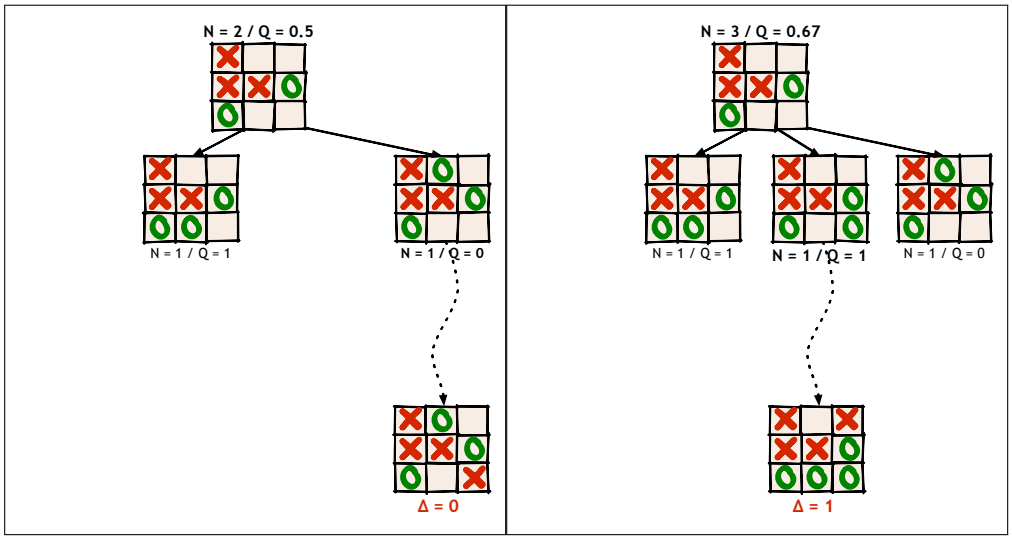
The Update policy

The reward Δ is back-propagated to the root, updating for each parent the number of visits $N(.)$ (incremented by 1) and the aggregation reward $Q(.)$ (the new proportion of wins).

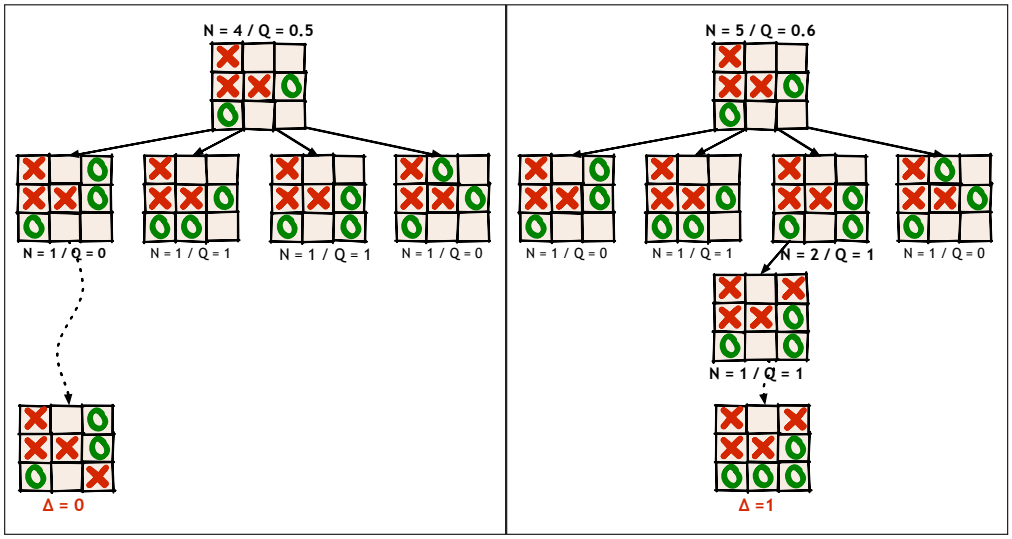
Example 12. *Figure 12 presents the first five iterations performed by the UCT algorithm for the TicTacToe game. Let us consider that the player p_1 plays against the player p_2. p_1 plays with the green circles and p_2 the red crosses. The current state of the game is given in Figure 12a. It is the player p_1's turn to play. The root of the tree is the current state. It is provided with two values: The number of times N it has been visited, and the proportion of wins Q. From this current state, the UCT algorithm can be run. The result of the first iteration, consisting of the four steps, is given in Figure 12b. During the first iteration, the method* Select *returns the root node (since it remains some of its children to expand). From the root node, i.e., the current state of the game, we randomly expand it with a not yet expanded child: The expanded node s_{exp} corresponds to the game state where p_1 plays in the middle cell of the third row. From this expanded node, a simulation is randomly played up to reach a terminal game state. In the case of the first iteration, the simulation reaches the terminal game state in which p_1 wins. The reward $\Delta = 1$ is back propagated through all the parents of s_{exp}: Thus the value N is incremented by one for all the parent nodes (since we visited these nodes one more time) and the value Q is set to 1 since the simulation leads to a win of p_1. Note that only the expanded node is stored within the tree, the nodes of the simulation are not stored in the tree. The second iteration consists in expanding the root node with another child that is not yet expanded in Figure 12c. For example, the child in which p_1 plays the cell in the middle of the first row. From this expanded node, a simulation is rolled out until a terminal state in which p_1 looses. The* Update *method back-propagates the reward by updating the value Q of the root node to 0.5 and increments the number of visits. Figure 12d and Figure 12e are related to the third and fourth iterations. Figure 12f is about the fifth iteration, but in this case, the* Select *method is a bit different since all the children of the root node are expanded. Thus, it is required to select one of these four children as the selected node s_{sel}. Due to the UCB measure, since all these children have been visited once, the exploration term of the UCB is the same for all the children. The exploitation term determines which child will be selected. Thus the algorithm selects randomly the second or the third child of the root node since they are the node that have led to a win. When the computational budget is reached, p_1 will*



(a) Initialisation. (b) Iteration #1.



(c) Iteration #2. (d) Iteration #3.



(e) Iteration #4. (f) Iteration #5.

Figure 12: 5 iterations of MCTS to the TicTacToe game.

play the action that leads to the best child of the current game state, i.e., with the child of the root node with the highest value for Q.

3.3 Improvements and issues

Monte Carlo tree search has motivated a lot of works for a couple of decades. The UCT algorithm proposes an implementation of a MCTS algorithm that enables to take into account the trade-off between the exploration of few visited parts of the search space and the exploitation of interesting areas by using a generalization of UCB1, namely UCT, in the SELECT method. The MCTS algorithms are aheuristic (i.e., without any knowledge), anytime (i.e., a solution is always available) and asymmetric (i.e., the expansion of the tree is based on the UCB). In the following, we present several improvements that have been proposed.

The first issue about MCTS is about the trade-off between exploration and exploitation in the enumeration of the search space. Indeed, as it is, MCTS tends to explore more than it exploits. This is partially due to the definition of the UCB but not only. In fact, UCB1 tends to give more importance of the exploration in the first visits of a node since the second term (exploration) of its formula is predominant w.r.t. the first term (exploitation). To tackle this issue, Auer et al. proposed an enhancement of UCB1, called *UCB1-Tuned*, that reduces the impact of the exploration term of the UCB1 formula [13]:

$$\textit{UCB1-Tuned}(s,s') = Q(s') + \sqrt{\frac{\ln N(s)}{N(s')} \times \min\left(\frac{1}{4}, \sigma^2(s') + \sqrt{\frac{2\ln N(s)}{N(s')}}\right)}$$

where $\sigma^2(s')$ is the variance of the rewards obtained by the child s' of the node s so far. Thus, the value exploration factor is at least divided by 2, and if the rewards obtained by the child s' are homogeneous, the impact of the exploration term is much more reduced. It means that even if an area of the search space has been lightly explored it is not required to explore it more if the rewards obtained so far during the simulation are similar: We may have a good estimation of the quality of this area. A second solution to tackle the over exploration with MCTS is to proceed to a progressive widening [45, 49]: It is not necessary to expand all the children of a node before selecting one of its children. In other words, when using progressive widening, the function SELECT(s) of Algorithm 1 is replaced by the function of Algorithm 2. This method is based on the following condition that states when it is required to expand the current selected node s based on the number of visits:

$$\lfloor \sqrt[b]{N(s)+1} \rfloor > \lfloor \sqrt[b]{N(s)} \rfloor$$

where b usually equals 2 or 4. If this condition holds, the node s is returned as the selected node s_{sel} and another child is expanded. If it does not hold, the SELECT policy recursively selects a child of s. In this way, it is possible to exploit interesting solutions before expanding (and so exploring) all the children of a node. Given enough time, all the children will be considered and the enumeration would converge to an exhaustive search as well. Now, the question is how to select the child to expand first. Usually, the enhancement RAVE (Rapid Action Value Estimator) is used to efficiently estimate which of the children is the most promising [73].

While the previous improvements are about the selection and the expansion of a node in the tree, now, we discuss about the back propagation of the rewards. Indeed, as presented in the pseudo code of Algorithm 1, the UPDATE method consists in back propagating both the reward and the number of visits from the expanded node s_{exp} to the root node. However, in the case of

Algorithm 2 The SELECT function using progressive widening.

```
1: function SELECT(s)
2:     while s is non-terminal do
3:         if ⌊∛(N(s)+1)⌋ > ⌊∛(N(s))⌋ then return s
4:         else s ← BESTCHILD(s)
5:         end if
6:     end while
7:     return s
8: end function
```

games (e.g., the Tic Tac Toe game), it is possible to update more than only the path from s_{exp} to the root node. For instance, in Figure 12d that presents the result of the third iteration, the final game state that is reached during the simulation shows that p_1 played in the middle cell of the third row before playing in the right cell of this row. Thus, it could have been interesting (and efficient) to not only update the root node and the expanded node of this iteration, but also the expanded node of the first iteration because this node is the case where p_1 plays in the middle cell of the third row. This enhancement is called All Moves As First (AMAF) and was introduced by Gelly and Silver [73]. AMAF treats all the actions played during the selection, the expansion and the simulation as if they were played in a selection step. In other word, a child of a node selected during the SELECT method will be updated if the action leading to this child has been played during the simulation. Besides, Permutation AMAF is an improvement of AMAF in which the update is done for all the paths that can eventually lead to the same final game state reached during the simulation [82]. Thus, not only some children are also updated (AMAF), but completely different paths in the tree are updated if they are permutations that can lead to the same final game state. Recently, Cazenave proposed the generalized rapid action value estimation that can use the AMAF values of a parent node if the current node has been lightly explored to have meaningful AMAF statistics [42].

The characteristics of the MCTS algorithms, where each iteration is independent w.r.t. the others, make this approach tractable for parallelisation. In this case, parallelisation can provide great advantages by performing several iterations over the tree in the same time, and thus be able to quickly and efficiently enumerate the search space. In the state of the art, parallelisation for MCTS has been studied from different points of view. First, we can proceed to several parallel simulations from the same expanded node in the tree. This is called leaf parallelisation [46] or at-the-leaves parallelisation [43]. This allows to sample more efficiently the subspace induced by the expanded node: Several rewards are back propagated and thus the estimation is more accurate. Second, a variant, called root parallelisation, was proposed to directly launch several MCTS in parallel [43, 46]. This enhancement is also called multi-tree MCTS. The information stored in the children of the root node of each MCTS tree is used to select the best action to play. The main advantage over the leaf-parallelisation is that each tree is independent and can be stopped anytime, whereas leaf-parallelisation has to wait for the longer simulation. Lastly, the so-called tree parallelisation enables to run several iterations on the same search tree [46]. Two variants of tree parallelisation have been proposed: The first consists in using a global mutex on the tree, whereas the second uses several local mutexes on each node of the tree. The first one is used when the simulation are time-consuming w.r.t. the traversal or the update of the tree. The latter is employed in the general case, when there is no guaranties on the run time of the simulations. However, the main drawback of tree parallelisation is that several threads can follow the same path in the tree since the method SELECT is based on the statistics of the nodes. To tackle this

problem, Chaslot et al. gives a temporary penalty to the nodes that are selected by a thread to avoid that they are selected once again by the following threads [46].

3.4 Applications

MCTS algorithms are mainly applied to games, and especially combinatorial games in which traditional methods of artificial intelligence fail. The game Go has become the new benchmark dataset to test AI algorithm [98]. Indeed, Go is a traditional 19x19 board game on which 2 players compete against each other. Its branching factor of 250 makes the game Go one of the most difficult board games to play. Go becomes one of the most used dataset to experiment with MCTS algorithms. The program MoGo is the first algorithm for Go that employs MCTS [72]. This algorithm uses RAVE to improve the efficiency of the exploration of the search tree. It was one of the best programs for playing Go. After that, several programs have been proposed. For instance, the algorithm Fuego, that also uses RAVE, beat a professional Go player. Recently, the Google's team Deep Mind has proposed the best program for the game Go, namely AlphaGo, that is partially based on MCTS [129]. This method combines the UCT algorithm and several deep learning tasks that enable to reduce the branching factor and the simulation length. This program beat the professionnal Go player Lee Sedol in March 2016 by winning four among the five games that have been played.

Gaudel and Sebag formalize feature selection as a one-player game and employ the UCT algorithm to solve it with their algorithm *FUSE* (Feature UCT Selection) [71]. This work aims at selecting the features from a feature space that are the more relevant w.r.t. the classification problem. For that, Gaudel and Sebag explore the powerset of the features (i.e., itemsets where the items are the features) with a MCTS method to find the sets of features that minimize the generalization error. Each node of the tree is a subset of feature, and each action consists of adding a new feature in the subset of features. The authors focus on reducing the high branching factor by using *UCB1-Tuned* and progressive widening with *RAVE* [73]. The aim of FUSE is thus to return the best subset of features (the most visited path of the tree), or to rank the features thanks to the RAVE score.

3.5 Conclusion

Monte Carlo tree search provides an efficient method to explore a huge search space with a high branching factor. The most famous algorithm, namely UCT, implements a variant of *UCB1* that enables to both explore lightly explored areas of the search space but also to exploit interesting solutions encountered so far. Based on this trade-off between the exploration and the exploitation, MCTS algorithms exhibit interesting characteristics:

- They are *aheuristic*: None prior knowledge on the domain is required. The pattern of the search space is learned incrementally with the number of iterations.
- They are *anytime*: A solution is always available and the result set keeps on improving with the time. The exploration converges to an exhaustive search if given enough time.
- The expansion of the tree is *asymmetric*: Based on the exploration/exploitation trade-off performed by the UCB, the tree is expanded in an asymmetric manner.

MCTS algorithms are efficient methods for combinatorial games. Recently, the famous AlphaGo algorithm proposed by the Google's team Deep Mind, beat professional Go players [129]. Despite

this highlighting all over the world, MCTS algorithms, to the best of our knowledge, have been seldomly used for other applications than combinatorial games. Gaudel and Sebag have proposed an adaptation of UCT to the combinatorial optimisation problem of feature selection [71], but we are not aware of previous attempts to adapt MCTS to pattern mining while it could improve significantly pattern enumeration methods. Indeed, the trade-off between the exploration of few visited parts of the search space and the exploitation of interesting solutions found so far can lead to the discovery of interesting patterns in large search space. Contrary to traditional sampling methods, MCTS algorithms are aheuristic, i.e., there is no need to provide a probability distribution on the pattern space since it is learned incrementally.

Chapter 4

Monte Carlo Tree Search for Pattern Mining

4.1 Introduction

Subgroup discovery (SD) is a formal framework that enables to elicit descriptive rules in label data [142]. One is given a set of objects associated to descriptions (that form a poset) and a mapping to one or several class labels. A subgroup is a description generalization whose discriminating ability is given by a quality measure (F1-score, accuracy, etc). In the last two decades, different aspects of SD have been widely studied: The description and target languages (quantitative, qualitative, etc.), the algorithms that enable the discovery of the best subgroups, and the definition of measures that express pattern interestingness. These three points are closely related. Many of the first approaches were *ad hoc* solutions lacking from easy implementable generalizations (see [119, 59] for surveys). SD hence faces two important challenges: How to define appropriate quality measures characterizing the singularity of a pattern; How to select an accurate heuristic search technique when an exhaustive enumeration of the pattern space is unfeasible.

In 2008, Lehman et al. introduced a more general framework called exceptional model mining (EMM, [104]) that tackles the first issue. EMM aims to find patterns that cover tuples that locally induce a model that substantially differs from the model of the whole dataset. This framework extends the classical SD settings and leads to a large class of models (i.e., quality measures) and applications, e.g., [139, 59, 17, 90]. In a similar fashion to other pattern mining approaches, SD and EMM select a heuristic search technique when exhaustive enumeration of the pattern space is unfeasible. The most widely used techniques are *beam search* [139, 112], *evolutionary* algorithms [54], and recently *pattern sampling* [114, 17]. The main goal of these heuristics is to drive the search towards the most interesting parts, i.e., the regions of the search space where patterns maximize a given quality measure (see Chapter 2). However, it often happens that the best patterns are redundant: They slightly differ in terms of their extent (dually intent). While several solutions have been proposed to filter out redundant subgroups, e.g., [139, 112], they do not discuss the choice of a beam search even though it causes this drawback. Indeed, a top-down level-wise greedy exploration of the patterns with a controlled level width that penalizes the diversity. Genetic algorithms and pattern sampling techniques are not greedy techniques in general, but the proposals in SD/EMM also face redundancy and diversity issues (see Chapter 2). Moreover, heuristic methods should converge to the result set of an exhaustive search (as many local optima as possible have to be found): They have to maximize the completeness of the result

set. Thus, there is still the need for implementing new SD/EMM heuristic exploration paradigms in a way that favors both the *patterns diversity* (i.e., with less redundancy as possible) and the completeness in the result set. This is the problem we propose to tackle in the present chapter through the use of Monte Carlo tree search (MCTS).

MCTS is a search method [36] mainly used in AI for domains, such as games and planning problems. The strength of MCTS is that it partially explores the search space building a tree in an incremental and asymmetric manner respecting the exploration/exploitation trade-off provided by the upper confidence bounds (UCB) [95]. MCTS is based on random simulations that are rolled out to explore the search space. Then, the expansion of the tree search depends on the rewards returned by simulations exploiting promising results but also exploring rarely visited parts of the tree. More importantly, the power of random search of MCTS leads to an *any-time mining* approach, in which a solution is always available, and which converges to an exhaustive search if given enough time and memory. MCTS quickly leads the search towards a diverse pattern set of high quality. In this context, our claims are the following:

- We formalize SD/EMM with an arbitrary pattern language and an arbitrary quality measure as a *single-turn single-player game*,
- We discuss how the four different existing methods of the UCT algorithm (Select, Expand, RollOut, Update) can be used for a simple description language (itemsets),
- We show how well known notions introduced in the pattern mining literature (DSF enumeration of closed patterns) can be incorporated in a MCTS: It requires to adapt the existing UCBs,
- We explain how our approach is agnostic of the pattern language and it can be instantiated for nominal, numerical and heterogeneous patterns,
- We provide an implementation of pattern mining based on Monte Carlo tree search, called mcts4dm,
- We empirically study our approach on artificial and benchmark data to assess both a better diversity and a high completeness in the result set. It is also where we compare mcts4dm with the main EMM/SD available algorithms.

The rest of this chapter is organized as follows. Section 4.2 reminds the main definitions of SD/EMM and highlights the problem of current search algorithms. Section 4.3 formally introduces SD and EMM as a single-turn single-player game when the patterns are itemsets, and details several ways rooted in pattern mining to solve it. This section also presents the slight adaption to consider numerical and heterogeneous patterns in general. Section 4.4 evaluates the capability of mcts4dm for efficiently finding the most diverse set of interesting patterns of various types. It is then successfully compared to well-known approaches of the literature in Section 4.5 before to conclude.

4.2 Subgroup Discovery

4.2.1 Definitions

We briefly recall the main definitions and issues of SD that are essential to make this section self-contained. The definitions of a label data and a subgroup are:

Table 7: Toy dataset

ID	a	b	c	$class(.)$
1	150	21	11	l_1
2	128	29	9	l_2
3	136	24	10	l_2
4	152	23	11	l_3
5	151	27	12	l_2
6	142	27	10	l_1

Definition 5 (Label dataset)**.** *Let $\mathcal{O}$, $\mathcal{A}$ and $\mathcal{C}$ be respectively a set of objects, a set of attributes, and a set of class labels. The domain of an attribute $a \in \mathcal{A}$ is $Dom(a)$ where a is either nominal or numerical. Each object is associated to a class label from $\mathcal{C}$ through $class : \mathcal{O} \mapsto \mathcal{C}$. $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$ is a label dataset.*

Definition 6 (Subgroup)**.** *The description of a subgroup is given by $d = \langle f_1, \ldots, f_{|\mathcal{A}|} \rangle$ where each f_i is a restriction on the value domain of the attribute $a_i \in \mathcal{A}$. The description of a subgroup corresponds to the pattern in the pattern mining setting. A restriction is either a subset of a nominal attribute domain, or an interval contained in the domain of a numerical attribute. The description d covers a set of objects called the extent of the subgroup, denoted $ext(d) \subseteq \mathcal{O}$, and its support is the number of objects in its extent and is defined by $supp(d) = |ext(d)|$. Note that, in this thesis, we denote $|S|$ the cardinality of the set S. For simplicity, a subgroup is either given by its intent, i.e., its description d, or by its extent $ext(d)$.*

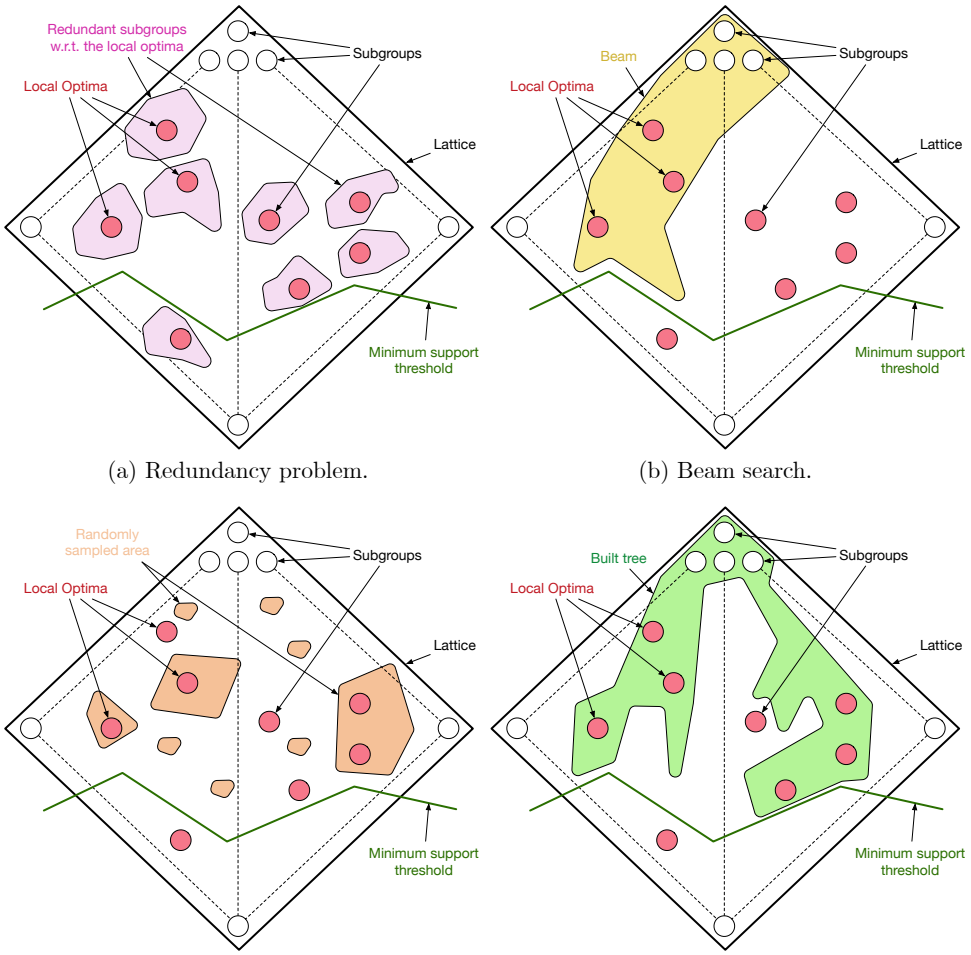
A quality measure φ is used to evaluate a subgroup. The quality measure reflects the difference between the model induced by the subgroup on the target attribute and the model induced by the entire label dataset. In SD, the model induced by a set of objects S is the proportion of objects of S associated to **one** class label $l \in \mathcal{C}$. The choice of the measure depends on the purpose of the application [68]. Usually, the Weighted Relative Accuracy (WRAcc) is used in SD [102]. Thus, we recall the SD problem:

Problem 2 (The SD problem)**.** *Given a label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a quality measure φ, a minimum support threshold $minSupp$ and an integer k, SD aims at extracting the top-k best frequent subgroups w.r.t. φ. The quality measure φ quantifies a deviation between the model induced by $ext(d)$ and $\mathcal{O}$, respectively.*

Note that, in the literature, the problem definition of SD can be slightly different from those we use in this section. Our definition is equivalent to those given in [59] in which we force that the set of constraints includes the minimum support constraint.

4.2.2 SD algorithmic issues

Definition 8 (Subgroup search space)**.** *The set of all descriptions is partially ordered and is structured as a lattice. We denote $s_1 \prec s_2$ and say that the subgroup s_1 is more specific than the subgroup s_2 if the description of s_1 is more specific than the one of s_2 w.r.t. the partial order (s_2 is more general than s_1). For instance, $\langle [23 \leqslant b \leqslant 29] \rangle$ is more general than $\langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$. Most of the SD/EMM algorithms exploit the lattice of subgroups.*



(a) Redundancy problem. (b) Beam search.

(c) Sampling exploration. (d) MCTS-based exploration.

Figure 6: Illustration of the different SD search algorithms.

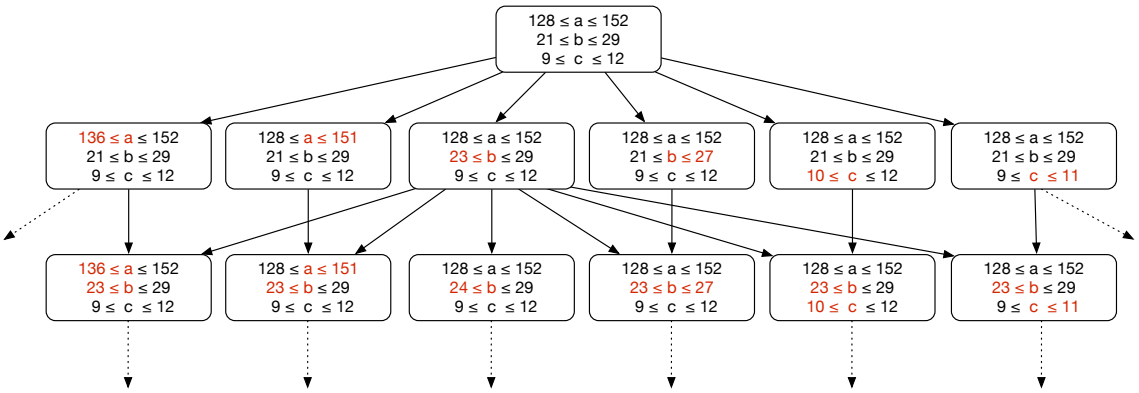


Figure 10: The upper part of the search space for data of Table 7.

Example 13. *To easily recover the figures, we recall the toy label dataset in Table 7 and the lattice given in Figure 10 in the following page. Note that, the domain of a numerical attribute is finite because during the exploration we only consider intervals whose bounds are the values taken in the data. Thus, for each numerical attribute, there are at most* $|\mathcal{O}|$ *different values, i.e.,*

a different one for each object. For each subgroup given by its description, there are 6 possible children: Increasing the lower bound or decreasing the upper bound for each numerical attribute. The top node is the more general description of the subgroups search space: Its extent is $\mathcal{O}$. In the following we consider the lattice of the subgroups as depicted in Figure 6. The nodes (circles) are the subgroups. The local optima (red circles) are the best subgroups w.r.t. the quality measure to extract. The minimum support threshold (green curve) delimits the border between the frequent subgroups and the others. Here, SD aims at extracting the result set containing all the best frequent subgroups (red circles above the green curve) and only them (depending on k).

Issue 1 (Redundancy/Diversity)**.** *The quality measure of a subgroup close to a local optimum s^* in the lattice is similar to – but lower than – the quality measure of s^*: A slight change in the description of a subgroup s close to s^* induces – in general – a slight change of the extent of s compared to those of s^*. It is desirable to avoid extracting the redundant subgroups close to a local optimum: This is the redundancy problem. Figure 6a depicts this phenomena where around each local optimum there is an area that contains redundant subgroups. Several solutions have been proposed for filtering the result set (or, e.g., the beam at each level of a beam search) [139] by using a similarity measure between patterns (on their support or descriptions) or using heuristic algorithms [35]. In our work, we will use the most popular similarity measure based on the similarity of the extents. Given two subgroups s_1 and s_2, the similarity measure between s_1 and s_2 is given by the Jaccard coefficient of their extent:*

$$sim(s_1, s_2) = \frac{|ext(s_1) \cap ext(s_2)|}{|ext(s_1) \cup ext(s_2)|}$$

Thus, the similarity constraint ensures that given a maximal similarity threshold $\Theta \in [0, 1]$, for all subgroups s_1, s_2 in the result set, $sim(s_1, s_2) \leqslant \Theta$.

Issue 2 (Completeness)**.** *When a dataset is too large, an exhaustive exploration of the search space is unfeasible. A heuristic such as beam search allows to explore only a little part of the search space based on a faster but greedy strategy (see Figure 6b). However, beam search comes also with its disadvantages: If some local optima are far away from the root node in the lattice, it is possible that the first levels leading to these optima are weak w.r.t. the quality measure, such that they will be ignored in the search. Sampling method is another popular heuristic approach used in SD/EMM (see Figure 6c). Samplings are based on a probability distribution over the subgroup space that gives more chance to an interesting subgroup to be drawn. Nevertheless, due to the long tail problem (many more subgroups are uninteresting than interesting) lots of uninteresting patterns might be drawn and interesting subgroups might be missed. Completeness entails the notion that* ***all optima should be present in the pattern set result****.*

One can now define the problem of pattern set enumeration in SD.

Problem 4 (Pattern set discovery)**.** *Given $\mathcal{D}(\mathcal{O}, \mathcal{A}, C, class)$, a quality measure φ, a minimum support threshold minSupp, a similarity measure sim, a maximum similarity threshold Θ, an integer k, compute a set of the top-k best patterns w.r.t. φ that has as little redundancy ($sim(s_1, s_2) \leqslant \Theta$, for all s_1, s_2 in the result set) and is as complete as possible.*

4.3 Pattern mining with MCTS

As previously stated, the problem of SD/EMM consists of finding the frequent patterns that maximize a quality measure φ. Let $\mathcal{S}$ be the set of all possible patterns. $\mathcal{S}$ is ordered thanks to a

specialization/generalization hierarchy, a poset $(\mathcal{S}, \prec)$ which is generally a lattice. The principle of solving the SD/EMM problem with a MCTS is based on the following:

- $\mathcal{S}$ corresponds to the set of game states. Each state, or pattern or node of the tree, is given an extent $ext(s)$, a support $supp(s)$ and a quality measure $\varphi(s)$. Moreover, we recall that the MCTS method requires that each node is provided with two values: The number of times the node s has been visited $N(s)$, and the average quality of the simulation walked through s so far $Q(s)$. The initial game state $s_0 \in \mathcal{S}$, root of tree, is the most general pattern.
- The actions for generating new game states are defined as pattern restrictions (for deriving pattern specializations/refinements).
- A simulation is a random sequence of actions, or pattern restrictions. A leaf is a maximal frequent pattern, i.e., any of its specializations is infrequent.

Designing a MCTS approach for a pattern mining problem is different than for a combinatorial game. The goal is not to decide, at each turn, what is the best action to play, but to explore the search space of patterns with the benefit of the exploitation/exploration trade-off and the tree that is stored in memory. Consequently, we consider SD/EMM as a *single-turn single-player game*.

For sake of simplicity, we consider in this section the most simple type of patterns: Itemsets. Obviously, most of our claims are given in the most generic form and can be slightly translated to consider other types of patterns. However, for some definitions, we discuss the case of nominal and numerical attributes. In our experiments, we deal jointly with itemsets, nominal and numerical attributes.

Problem 5 (EMM when descriptions are itemsets)**.** *Let $\mathcal{I}$ be a set of items. A transaction is a subset of items $t \subseteq \mathcal{I}$. A transaction database is a set of transactions $\mathcal{T} = \{t_1, ..., t_n\}$. An itemset is an arbitrary subset of items $p \subseteq \mathcal{I}$. Its extent is given by $ext(p) = \{t \in \mathcal{T} | p \subseteq t\}$, and its support is $supp(p) = |ext(p)|$. Its evaluation measure $\varphi(p)$ depends on the quality measure chosen in the EMM instance. The problem is to find the best itemsets w.r.t φ.*

It follows that the search space is given by $\mathcal{S} = (2^{\mathcal{I}}, \subseteq)$. The initial pattern is the empty set: $s_0 = \varnothing$. The actions that lead to specializations, or supersets, are the items $\mathcal{I}$. A simulation is a random sequence of item additions. We are now able to specify the strategies of the four steps of the UCT algorithm especially tuned for a SD/EMM problem. Figure 6d is the exploration performed by the MCTS in the lattice of subgroups. Due to the exploration/exploitation trade-off embedded by MCTS, it suggests both a high diversity and a high completeness in the result set if given enough budget (i.e., enough iterations).

4.3.1 The Select method

The Select method consists of selecting the most promising node s_{sel} in terms of exploration vs. exploitation. For that, the well-known UCT and UCB1 (UCT with $C_p = 1/2$) can be used in our settings. However, more sophisticated UCBs have been designed for single player games. The single-player MCTS (SP-MCTS [128]) adds a third term to the UCB to take into account the variance σ^2 of the rewards obtained by the child so far. SP-MCTS of a child s' of a node s is:

$$SP\text{-}MCTS(s, s') = Q(s') + C\sqrt{\frac{2 \ln N(s)}{N(s')}} + \sqrt{\sigma^2(s') + \frac{D}{N(s')}}$$

where the constant C is used to weight the exploration term (it is fixed to 0.5 in its original definition [128]) and the term $\frac{D}{N(s')}$ inflates the standard deviation for infrequently visited children (D is also a constant). In this way, the reward of a rarely visited node is considered as less certain: It is still required to explore it to get a more precise estimate of its variance. If the variance is still high, it means that the subspace from this node is not homogeneous w.r.t. the quality measure and then further exploration is needed.

Also, *UCB1-Tuned* [13] is designed to reduce the impact of the exploration term of the original UCB1 by weighting it with either an approximation of the variance of the rewards obtained so far or the factor 1/4. UCB1-Tuned of a child s' of s is:

$$UCB1\text{-}Tuned(s, s') = Q(s') + \sqrt{\frac{\ln N(s)}{N(s')} \min\left(\frac{1}{4}, \sigma^2(s') + \sqrt{\frac{2 \ln N(s)}{N(s')}}\right)}$$

If required, the pattern evaluation measure φ can be normalized (e.g., for UCT).

4.3.2 The Expand method

The Expand step consists in adding a new node in the search tree. In the following, we present different refinement operators, and how to avoid duplicate nodes in the search tree.

The refinement operators

The simple way to expand the selected node s_{sel} is to choose uniformly an available attribute w.r.t. s_{sel}, that is to specialize s_{sel} into s_{exp} such that $s_{exp} \prec s_{sel}$: s_{exp} is a refinement of s_{sel}.

Definition 9 (Refinement operator)**.** *A refinement operator is a function $ref : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ that derives from a pattern s a set of more specific patterns $ref(s)$ such that:*

- *(i)* $\forall s' \in ref(s), s' \prec s$
- *(ii)* $\forall s'_i, s'_j \in ref(s), i \neq j, s'_i \not\preceq s'_j, s'_j \not\preceq s'_i$

The extent of a refined pattern $s' \in ref(s)$ is included in the extent of the pattern s. Formally, $\forall s_i \in ref(s), ext(s') \subseteq ext(s)$. Then, $supp(s') \leqslant supp(s)$. This is a basic property known as anti-monotonicity of the support.

In other words, a refinement operator gives to any pattern s a set of its specializations, that are pairwise incomparable (an anti-chain). The *refine* operation can be implemented in various ways given the kind of patterns we are dealing with. Most importantly, it can return all the direct specializations only to ensure that the exploration will, if given enough budget, explore the whole search space of patterns. Furthermore, it is unnecessary to generate infrequent patterns.

Definition 10 (Direct-refinement operator)**.** *A direct refinement operator is a refinement operator $directRef : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ that derives from a pattern s the set of direct more specific patterns s' such that:*

- *(i)* $\forall s' \in directRef(s)$, $s' \prec s$
- *(ii)* $\nexists s'' \in \mathcal{S}$ *s.t.* $s' \prec s'' \prec s$
- *(iii) For any* $s' \in directRef(s)$, s' *is frequent, that is* $supp(s') \geqslant minSupp$

Example 14. *First, let us consider EMM when descriptions are itemsets and a pattern s. The direct-refinement operator is given by $directRef(s) = \{s \cup i \mid i \in (\mathcal{I} \backslash s)\}$. There are at most $|\mathcal{A}| = |\mathcal{I}|$ children for a pattern s in the search tree. Second, when dealing with nominal attributes, for each nominal attribute $a \in \mathcal{A}$, we can derive $|Dom(a)|$ different children: One for each available value of the attribute a. For a pattern s, the direct-refinement operator is $directRef(s) = \{s \cup [a_i = v_j] \mid a_i \in (\mathcal{A} \backslash Attr(s)) \wedge v_j \in Dom(a_i)\}$, where $Attr(s)$ is the set of attributes already used in the effective restrictions of s. Finally, for numerical attributes, we can derive two direct-refined subgroups for each numerical attribute: Applying the minimal left change (increasing the lower bound of the interval to the next higher value taken in the data) and the minimal right change (decreasing the upper bound of the interval to the next lower value taken in the data) [88]. Thus, there are at most $2 \times |\mathcal{A}|$ children for a pattern s in the search tree.*

Definition 11 (The *direct-expand* strategy)**.** *We define the direct-expand strategy as follows: From the selected node s_{sel}, we randomly pick a – not yet expanded – node s_{exp} from $directRef(s_{sel})$ and we add it in the search tree.*

Below, we propose two different strategies that address the usual problem in pattern mining: A pattern s_{sel} can be expanded into a node s_{exp} with the same extent. Most quality measures φ used in SD and EMM are solely based on the extent of the patterns [59, 90]. However, with the *direct-refinement operator*, a large number of tree nodes may have the same extent (and thus the same support) as their parent, hence the same quality measure value. This redundancy may bias the exploration and more iterations will be required. For that, we propose to use the notion of closed patterns and their generators. The following definition overrides Definition 2 and Definition 3 considering the generic lattice structure $(\mathcal{S}, \prec)$:

Definition 12 (Closed descriptions and their generators)**.** *The equivalence class of an pattern s is given by $[s] = \{s' \in \mathcal{S} \mid ext(s) = ext(s')\}$. Each equivalence class has a unique smallest element w.r.t. $\prec$ that is called the closed pattern: s is said to be closed iff $\nexists s'$ such that $s' \prec s$ and $ext(s) = ext(s')$. The non-closed patterns are called generators.*

Definition 13 (Generator-refinement operator)**.** *A generator refinement operator is a refinement operator $genRef : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ that derives from a pattern s the set of more specific patterns s' such that:*

- *(i) $\forall s' \in genRef(s)$, $s' \notin [s]$*
- *(ii) $\nexists s'' \in \mathcal{S} \backslash genRef(s)$ s.t.*
  - *$s'' \notin [s]$,*
  - *$s'' \notin [s']$, and*
  - *$s' \prec s'' \prec s$*
- *(iii) For any $s' \in genRef(s)$, s' is frequent, that is $supp(s') \geqslant minSupp$*

Example 15. *For itemsets and nominal attributes, the generator-refinement operator of a pattern s is given by: $genRef(s) = \{s' \in directRef(s) \mid supp(s') \neq supp(s)\}$. In other words, with the generator-refinement operator, the children of a pattern s are the direct-refined patterns that are not in the same equivalence class than s. For numerical attributes, the generator-refinement operator still derives two refined patterns for each attribute: Iteratively applying minimal left (resp. right) changes until the support changes.*

Definition 14 (The *gen-expand* strategy)**.** *To avoid the exploration of patterns with the same extent in a branch of the tree, we define the gen-expand strategy as follows: From the selected node s_{sel}, we randomly pick a – not yet expanded – refined pattern from $genRef(s_{sel})$, called s_{exp}, and add it to the search tree.*

Besides, when facing a SD problem whose aim is to characterize a label $l \in \mathcal{C}$ we can implement an extension of the previous refinement operator based on generators on the extents of both the subgroup and the label.

Definition 15 (Label-refinement operator)**.** *A label refinement operator w.r.t. $l \in \mathcal{C}$ is a refinement operator $labRef : \mathcal{S} \to 2^{\mathcal{S}}$ that derives from a pattern s the set of more specific patterns s' such that:*

- *(i) $\forall s' \in labRef(s)$, $ext_l(s') \neq ext_l(s)$, with $ext_l(s) = \{o \in ext(s) \mid class(o) = l\}$*
- *(ii) $\nexists s'' \in \mathcal{S} \backslash labRef(s)$ s.t.*
  - *$ext_l(s'') \neq ext_l(s)$,*
  - *$ext_l(s'') \neq ext_l(s')$, and*
  - *$s' \prec s'' \prec s$*
- *(iii) For any $s' \in labRef(s)$, s' is frequent, that is $supp(s') \geqslant minSupp$*

Example 16. *For itemsets and nominal attributes, the label-refinement operator of a pattern s w.r.t. the label $l \in \mathcal{C}$ is given by: $labRef(s) = \{s' \in genRef(s) \mid supp_l(s') \neq supp_l(s)\}$. For numerical attributes, the label-refinement operator still derives two refined patterns for each attribute: Iteratively applying minimal left (resp. right) changes until the number of objects in the extent associated to l is changed.*

Definition 16 (The *label-expand* strategy)**.** *From the selected node s_{sel}, when facing a SD problem, the label-expand strategy consists in randomly picking a – not yet expanded – refined pattern from $labRef(s_{sel})$, called s_{exp}, and adding it to the search tree.*

Avoiding duplicates in the search tree

Previously, we defined several refinement operators to avoid the redundancy within a branch of the tree, i.e., do not expand s_{sel} with a pattern whose extent is the same because the quality measures φ will be equal. However, another redundancy issue remains at the tree scale. Indeed, since the pattern search space is a lattice, a pattern can be generated in nodes in different branches of the Monte Carlo tree, that is, with different sequences of actions. For example, with $\mathcal{I} = \{a, b, c\}$, all permutations of the sequence $\langle a, b, c\rangle$ could be generated. As such, it will happen that a part of the search space is sampled several times in different branches of the tree. However, the visit count $N(s)$ of a node s will not count visits of other nodes that depict exactly the same pattern: The UCB is biased. To tackle this aspect, we implement two methods: (i) Using a lectic order or (ii) detecting and unifying the duplicates within the tree. These two solutions can be used for any refinement operator. Note that, enabling both these solutions is useless since each of them ensures to avoid duplicates within the tree.

Lectic order (LO). A solution is to generate each pattern only once. The visit count should be then adapted accordingly. First we detail the enumeration procedure. Then, we show how to adapt the visit count in the computation of the UCB.

Pattern enumeration without duplicate is at the core of constraint-based pattern-mining [33]: The goal of exhaustive search is to output the correct, complete and non-redundant collection of patterns. Most of the time, such a non-redundant exploration is based on a total order on the set of attribute restrictions. This poset is written by $(A, \preccurlyeq)$.

Example 17. *For itemset patterns, $A = \mathcal{I}$ and a lectic order is chosen on $\mathcal{I}$. Usually, it is the lexicographic order, e.g., $a \prec b \prec c \prec d$ for $A = \{a, b, c, d\}$: Thus, for example, $bc \prec ad$. A complete and non redundant enumeration of all patterns is then direct. Consider that a node s has been generated with a restriction a_i: We can expand the node only with restrictions a_j such that $a_i \prec a_j$. This total order also holds for numerical attributes by considering the minimal changes (see the work of Kaytoue et al. for further details [88]).*

We can use this technique to enumerate the lattice with a depth-first search (DFS), which ensures that each element of the search-space is visited exactly once. An example is given in Figure 13. However, it induces a strong bias: An MCTS algorithm would sample this tree instead of sampling the pattern search space. In other words, a small restriction w.r.t. $\prec$ has much less chances to be picked than a largest one. Going back to the example in Figure 13 (middle), the item a can be draw only once through a complete DFS; b twice; while c four times (in bold). It follows that patterns on the left hand side of the tree have less chances to be generated, e.g., $prob(\{a, b\}) = 1/6$ while $prob(\{b, c\}) = 1/3$. These two itemsets should however have the same chances to be picked as they have the same size. This disequilibrium can be corrected by weighting the visit counts in the UCT with the normalized exploration rate (see Figure 13 (right)).

Definition 17 (Normalized exploration rate)**.** *Let $\mathcal{S}$ be the set of all possible patterns. The normalized exploration rate of a pattern s is:*

$$\rho_{norm}(s) = \frac{V_{total}(s)}{V_{lectic}(s)} = \frac{|\{s' | s' \prec s \in \mathcal{S}\}|}{|\{s' | s \prec s' \wedge s' \prec s \in \mathcal{S}\}|}$$

From this normalized exploration rate we can adapt the UCBs when enabling the lectic order. For example, we can define the *DFS-UCT* of a child s' of a pattern s derived from the *UCT* as follows:

$$DFS\text{-}UCT(s, s') = Q(s') + 2C_p\sqrt{\frac{2\ln\left(N(s) \cdot \rho_{norm}(s)\right)}{N(s') \cdot \rho_{norm}(s')}}$$

Proposition 1 (Normalized exploration rate for itemsets)**.** *For itemsets, let s_i be the child of s obtained by playing action a_i and i is the rank of a_i in $(A, \preccurlyeq)$: $\rho_{norm}(s_i) = \frac{2^{(|\mathcal{I}|-|s_i|)}}{2^{(|\mathcal{I}|-i-1)}}$.*

Proof. Let $V_{lectic}(s_i)$ be the size of the search space sampled under s_i using a lectic enumeration, and $V_{total}(s_i)$ be the size of the search space without using a lectic enumeration. The exploration rate is thus given by $\rho(s_i) = \frac{N(s_i) \times V_{total}(s_i)}{V_{lectic}(s_i)}$. Thus, $\rho_{norm}(s_i) = \frac{V_{total}(s_i)}{V_{lectic}(s_i)}$ where $V_{total}(s_i) = 2^{(|\mathcal{I}|-|s_i|)}$ and $V_{lectic}(s_i) = 2^{(|\mathcal{I}|-i-1)}$ for itemsets. □

Proposition 2 (Normalized exploration rate for a numerical attribute)**.** *For a single numerical attribute a, $\rho_{norm}(.)$ is defined as follows :*

- *Let $s' = \langle \alpha_i \leqslant a \leqslant \alpha_j \rangle$ obtained after a left change: $\rho_{norm}(s') = 1$.*
- *Let $s' = \langle \alpha_i \leqslant a \leqslant \alpha_j \rangle$ obtained after a right change. Let n be the number of values from $Dom(a)$ in $[\alpha_i, \alpha_j]$: $\rho_{norm}(s') = \frac{n+1}{2}$.*

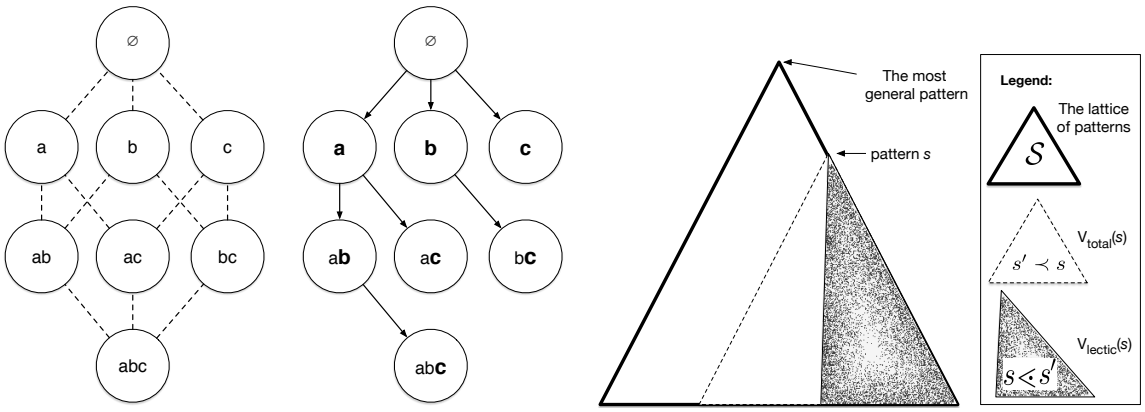


Figure 13: Search space as a lattice (left), DFS of the search space (middle), and the principles of the normalized exploration rate.

Proof. As explained in the proof of the DFS-UCT for itemsets (Proposition 1), $\rho_{norm}(s) = \frac{V_{total}(s)}{V_{lectic}(s)}$. For a numerical attribute, $V_{total}(s) = n(n+1)/2$, i.e. the number of all sub intervals. If s was obtained after a left change, $V_{lectic}(s) = n(n+1)/2$ as both left and right changes can be applied. If s was obtained after a right change, $V_{lectic}(s) = n$, as only n right changes can be applied. It follows that $\rho_{norm}(s) = \frac{n(n+1)/2}{n(n+1)/2} = 1$ if s was obtained from a left change and $\rho_{norm}(s) = \frac{n(n+1)/2}{n} = \frac{n+1}{2}$ otherwise. □

Permutation unification (PU). The permutation unification is a solution that allows to keep a unique node for all duplicates of a pattern that can be expanded within several branches of the tree. This is inspired from *Permutation AMAF*, a method used in traditional MCTS algorithm to update all the nodes that can be concerned by a play-out [82]. A unified node no longer has a single parent but a list of all duplicates' parent. This list will be used when back-propagating a reward.

This is made precise in Algorithm 3. Consider that the node s_{exp} has been chosen as an expansion of the selected node s_{sel}. The tree generated so far is explored for finding s_{exp} elsewhere in the tree: If s_{exp} is not found, we proceed as usual; otherwise s_{exp} becomes a pointer to the duplicate node in the tree. In our MCTS implementation, we will simply use a hash map to store each pattern and the node in which is has been firstly encountered.

Algorithm 3 The permutation unification principle.

```
1: H ← new Hashmap()
2: function EXPAND(s_sel)
3:     randomly choose s_exp from non expanded children of s_sel
4:     if (node ← H.get(s_exp)) ≠ null then
5:         node.parents.add(s_sel)
6:         s_exp ← node
7:     else
8:         s_exp.parents ← new List()
9:         s_exp.parents.add(s_sel)
10:        H.put(s_exp, s_exp)          ▷ A pointer on the unique occurrence of s_exp
11: end if
12: add new child s_exp to s_sel in the tree          ▷ Expand s_sel with s_exp
13: return s_exp
14: end function
```

4.3.3 The RollOut method

From the expanded node s_{exp} a simulation is run (RollOut). With standard MCTS, a simulation is a random sequence of actions that leads to a terminal node: A game state from which a reward can be computed (win/loss). In our settings, it is not only the leaves that can be evaluated, but any pattern s encountered during the simulation is given with its quality measure $\varphi(s)$. Thus, we propose to define the notion of path (the simulation) and reward computation (which nodes are evaluated and how these different rewards are aggregated) separately.

Definition 18 (Path Policy)**.** *Let s_1 the node from which a simulation has to be run (i.e., $s_1 = s_{exp}$). Let $n \geqslant 1 \in \mathbb{N}$, we define a path $p(s_1, s_n) = \{s_1, \ldots, s_n\}$ as an ordered list of patterns starting from s_1 and ending with s_n such that: $\forall i \in \{1, \ldots, n-1\}, s_{i+1}$ is a refined pattern of s_i. We denote $\mathcal{P}(s_1, s_n)$ the set of all possible paths from s_1 to s_n.*

- *naive-roll-out: A path length n is randomly picked in $(1, ..., pathLength)$ where $pathLength$ is given by the user ($pathLength = |\mathcal{I}|$ by default) using the direct refinement operator.*
- *direct-freq-roll-out: The path is extended with a randomly chosen restriction until it meets an infrequent pattern s_{n+1} using the direct refinement operator. s_n is a leaf of the tree in our settings.*
- *large-freq-roll-out overrides the direct-freq-roll-out by using specializations that are not necessarily direct. Several actions are added instead of one to create a new element of the path. The number of added actions is randomly picked in $(1, ..., jumpLength)$ where $jumpLength$ is given by the user ($jumpLength = 1$ gives the previous policy). A jump of m direct refinements only requires to explore once the data, whereas m successive direct refinements require to explore m times the data. Thus, this policy will drastically reduce the simulation times yet still properly exploring parts of the search space.*

Definition 19 (Reward Aggregation Policy)**.** *Let s_1 the node from which a simulation has been run and $p(s_1, s_n)$ the associated random path. Let $\mathcal{E} \subseteq p(s_1, s_n)$ be the subset of nodes to be evaluated. The aggregated reward of the simulation is given by: $\Delta = aggr(\{\varphi(q) \forall q \in \mathcal{E}\}) \in [0; 1]$ where $aggr$ is a aggregation function. We define several reward aggregation policies:*

- *terminal-reward: $\mathcal{E} = \{s_n\}$ and $aggr$ is the identity function.*
- *random-reward: $\mathcal{E} = \{s_i\}$ with a random $1 \leqslant i \leqslant n$ and $aggr$ the identity function.*
- *max-reward: $\mathcal{E} = p(s_1, s_n)$ and $aggr$ is the $max(.)$ function*
- *mean-reward: $\mathcal{E} = p(s_1, s_n)$ and $aggr$ is the $mean(.)$ function.*
- *top-k-mean-reward: $\mathcal{E} = $ top-k$(p(s_1, s_n))$, $aggr$ is the $mean(.)$ function and top-k(X) returns the k elements with the highest φ.*

A basic MCTS forgets any state encountered during a simulation. This is not optimal for single player games [22]: A pattern with a high φ should not be forgotten as we might not expand the tree enough to reach it. We propose several memory strategies for our concern.

Definition 20 (Roll-out Memory Policy)**.** *A roll-out memory policy specifies which of the nodes of the path $p = (s_1, s_n)$ shall be kept in an auxiliary data structure M.*

- *no-memory: Any pattern in $\mathcal{E}$ is forgotten.*
- *all-memory: All evaluated patterns in $\mathcal{E}$ are kept.*
- *top-k-memory: A list M stores the best k patterns in $\mathcal{E}$ w.r.t. $\varphi(.)$.*

4.3.4 The UPDATE method

The back-propagation method updates the tree according to a simulation. Let s_{sel} be the selected node and s_{exp} its expansion from which the simulation is run: This step aims at updating the estimation $Q(.)$ and the number of visits $N(.)$ of each parent of s_{exp} recursively. Note that s_{exp} may have several parents when we enable the permutation unification. The number of visits $N(.)$ is always incremented by one. We consider three ways of updating $Q(.)$:

- *mean-update*: $Q(.)$ is the average of the rewards Δ back-propagated through the node so far (basic MCTS).
- *max-update*: $Q(.)$ is the maximum reward Δ back-propagated through the node so far. This strategy allows to identify a local optimum within a part of the search space that contains most of uninteresting patterns. Thus, it gives more chance for this area to be exploited in the next iterations.
- *top-k-mean-update*: $Q(.)$ average of the k best rewards Δ back-propagated through the node so far. It gives a stronger impact for the parts of the search space containing several local optima.

We introduced the *max-update* and *top-k-mean-update* policies as it may often happen that high-quality subgroups are rare and scattered in the search space. The mean value of rewards from simulations would converge towards 0 (there are too many low quality subgroups), whereas the maximum value (and top-k average) of rewards enables to identify the promising parts of the search space.

4.3.5 Search end and result output

Once the computational budget is reached, or when the tree is fully expanded, the search is stopped. The number of tree nodes equals the number of iterations that have been performed. It remains now to explore this tree and the data structure M built by the memory policy to output the list of diverse and non-redundant subgroups.

Let $\mathcal{P} = T \cup M$ be a pool of patterns, where T is the set of patterns stored in the nodes of the tree. The set $\mathcal{P}$ is totally sorted w.r.t. φ in a list ι. Thus, we have to pick the k-best diverse and non-redundant subgroups within this huge pool of nodes ι to return the result set of subgroups $\mathcal{R} \subseteq \mathcal{P}$. For that, we develop a post-processing approach that filters out redundant subgroups from the diverse pool of patterns ι based on the similarity measure sim and the maximum similarity threshold Θ. Recursively, we poll (and remove) the best subgroup s^* from ι, and we add s^* to $\mathcal{R}$ if it is not redundant with any subgroup in $\mathcal{R}$.

This post processing for removing redundancy is often used in the SD/EMM literature [139, 112]. It requires however that the pool of patterns has a reasonable cardinality which may be problematic with MCTS. The allowed budget always enables such post-processing in our experiments (up to one million iterations).

4.4 How to setup MCTS4DM?

We first present datasets used for the evaluation of SD and EMM methods in the literature and we propose an artificial data generator that allows to generate datasets with a controlled insertion of interesting patterns. We then experiment with our approach and we report its effectiveness. The experiments were carried out on an Intel Core i7 CPU 2.2 Ghz machine with 16 GB RAM

Name	# Objects	# Attributes	Type of attributes	Target attribute
BreastCancer	699	9	Numeric	Benign
Cal500	502	68	Numeric	Angry-Agressive
Emotions	594	72	Numeric	Amazed-suprised
Ionosphere	352	35	Numeric	Good
Iris	150	4	Numeric	Iris-setosa
Mushroom	8,124	22	Nominal	Poisonous
Nursery	12,961	8	Nominal	class=priority
TicTacToe	958	9	Nominal	Positive
Yeast	2,417	103	Numeric	Class1

Table 9: Benchmark datasets experimented on in the SD and EMM literature.

running under macOS Sierra. To ensure the reproducibility of our results, we made our code and data publicly available[7]. Our MCTS implementation for pattern mining, called MCTS4DM is written in Java and is provided with a configuration file to choose any of the strategies introduced in this chapter.

4.4.1 Artificial data generator and benchmark datasets

The main goal of this work is to propose a heuristic approach that is able to output a diverse set of patterns. To evaluate our approach, a ground-truth is needed: All subgroups of a dataset shall be known beforehand such that heuristic search result could be then evaluated. Such a ground-truth is however available only if one is able to run and terminate an exhaustive search, which limits the type of data to be experimented with. Accordingly, we propose (i) to take into account benchmark data for which an exhaustive search is possible, and (ii) to generate artificial data in which interesting subgroups are hidden in noise and other frequent patterns (i.e., a ground-truth is known). In Section 4.5.3 we present a large real life dataset to assess the efficiency of MCTS4DM when an exhaustive search is not tractable.

Firstly, we gathered the benchmark datasets used in the recent literature of SD and EMM [139, 57, 137, 138, 60]. Table 9 lists them, mainly taken from the UCI repository, and their properties. Secondly, we adapted the artificial data generator given in [90] in which the authors considered an EMM task where the target is a subgraph induced by a subgroup. We modify their generator to produce a dataset with nominal attributes and a binary target (positive and negative class labels).

More specifically, the generator takes the parameters given in Table 10 and works as follows. A numerical object/attribute data table is generated with an additional binary attribute: The target taking a positive or negative value. The number of objects, attributes and attributes values are controlled with the parameters nb_obj, nb_attr and $domain\_size$. Our goal is to hide $nb\_patterns$ patterns in noise, so we generate random descriptions of random lengths $Ground = \{d_i \mid i \in [1, nb\_patterns]\}$. For each pattern, we generate $pattern\_sup$ objects positively labeled with a probability of $1 - noise\_rate$ to be covered by the description d_i, and $noise\_rate$ for not being covered. We also add $pattern\_sup \times out\_factor$ negative examples for the pattern d_i: It will allow patterns with different quality measures (here the WRAcc). Finally, we add random objects until we reach a maximum number of transactions nb_obj.

4.4.2 Experimental framework

We perform a large pool of experiments to assess this new exploration method for pattern mining. For that, we have designed an experimental framework that enables to test the different

[7] https://github.com/guillaume-bosc/MCTS4DM

Name	Description	$\mathcal{P}_{small}$	$\mathcal{P}_{medium}$	$\mathcal{P}_{large}$
nb_obj	Number of objects	2,000	20,000	50,000
nb_attr	Number of attributes	5	5	25
$domain\_size$	Domain size per attribute	10	20	50
$nb\_patterns$	Number of hidden patterns	3	5	25
$pattern\_sup$	Support of each hidden pattern	100	100	100
out_factor	Proba. of a pattern labeled −	0.1	0.1	0.1
$noise\_rate$	Proba. of a object to be noisy	0.1	0.1	0.1

Table 10: Parameters of the artificial data generator.

combinations of factors for all the strategies we introduced in previous sections. Each experiment are run on the nine benchmark datasets. An experiment consists in varying a unique strategy parameter while the others are fixed. Since MCTS4DM uses random choices, each experiment is run five times and only the mean of the results is discussed.

The default parameters. For each benchmark dataset, we provide a set of default parameters. Indeed, due to the specific characteristics of each dataset, a common set of default parameters is unsuitable. Table 10 displays the default parameters for each dataset. However, all datasets share a subset of common parameters:

- The maximum size of the result set is set to $maxOutput = 50$.
- The maximum similarity threshold is set to $\Theta = 0.7$.
- The maximum description length is set to $maxLength = 5$.
- The quality measure used is $\varphi = WRAcc$. Note that for each dataset, the experiments consist in finding subgroups deviating on the first target label.
- The SP-MCTS is used as the default UCB.
- The permutation unification (PU) strategy is used by default.
- The refinement operator for the EXPAND method is set to *tuned-min-gen-expand*.
- The *direct-freq-roll-out* strategy is used for the ROLLOUT method
- The reward aggregation policy is set to *max-reward*.
- The memory policy is set to *top-1-memory*.
- The update policy is set to *max-udpate*.

The list of experiments. Evaluating MCTS4DM is performed by means of seven different batches of experiments:

- Section 4.4.3 is about the choice of UCB.
- Section 4.4.4 deals with the several strategies for EXPAND .
- Section 4.4.5 presents the leverage of all the possibilities for ROLLOUT.
- Section 4.4.6 shows out the impact of the MEMORY strategy.

- Section 4.4.7 compares the behaviors of all the strategies for UPDATE.
- Section 4.4.8 performs the experiments when varying the computational budget, namely the number of iterations.
- Section 4.4.9 experiments with the completeness of MCTS4DM on artificial data when varying the number of iterations.

For simplicity and convenience, for each experiment we display the same batch of figures. For each dataset we show (i) the boxplots of the quality measure φ of the subgroups in the result set, (ii) the histograms of the runtime and (iii) the boxplots of the description length of the subgroups in the result set depending on the strategies that are used. In this way, the impacts of the strategies are easy to understand.

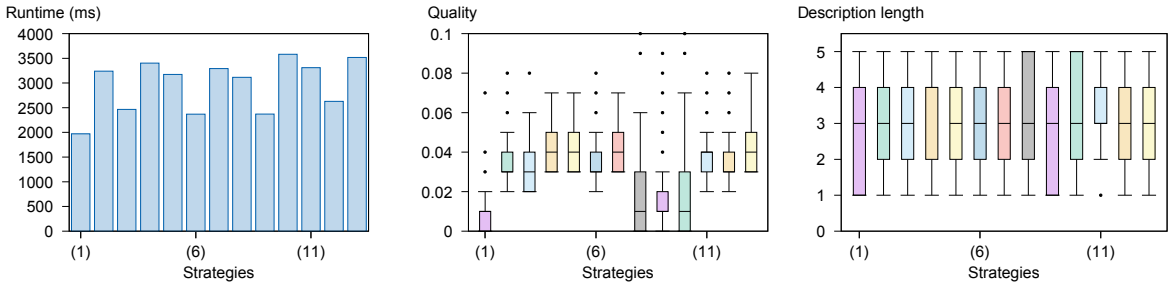
4.4.3 The SELECT method

The choice of the UCB is decisive, because it handles the exploration / exploitation trade-off. Indeed, the UCB chooses which part of the search tree will be expanded and explored. We presented four existing UCBs and we introduced a new one based on the *normalized exploration rate* to take into account a partial order on the subgroups. Experimenting with these different UCBs has to be done when selecting the exploration method. In fact, we showed that there exists different nodes within the search tree that are exactly the same ones (the search space is explored as a tree but it is a lattice). Handling such a redundancy can be done with either *permutation unification* (PU) or using a *lectic order* (LO). Thus, Figure 14(bottom) presents all the strategies we experimented. Comparing the runtime for all the strategies leads to conclude that there is no impact when computing the several UCBs (see Figure 14(a)). Indeed, the impact of the UCBs lies in its computation. However, we can notice that when LO is used, the runtime is lower. This result is expected because with LO, the search space is less large since there is no redundant subgroups.

Figure 14(b) depicts the boxplots of the quality measure of the result set when varying the UCB. The results suggest that the *UCB1-Tuned* and *DFS-UCT* lead to weaker quality results for several datasets: On the *Cal550*, *Emotions* and *Yeast* datasets, the quality measures of the result sets are worse than the results of other UCBs (see, e.g., Figure 14(b)). This is due to the fact that the search space of these datasets is larger than the others with many local optima, and the *UCB1-Tuned* is designed to explore less, thus less local optima are found. Besides, the *SP-MCTS* seems to be more suitable for SD problems: The quality is slightly better than other UCBs for

Dataset	minSupp	# iterations	Path Policy
BreastCancer	10	50k	*large-freq-roll-out* ($jumpLength = 30$)
Cal500	10	100k	*large-freq-roll-out* ($jumpLength = 30$)
Emotions	10	100k	*large-freq-roll-out* ($jumpLength = 30$)
Ionosphere	10	50k	*large-freq-roll-out* ($jumpLength = 30$)
Iris	10	50k	*large-freq-roll-out* ($jumpLength = 30$)
Mushroom	30	50k	*direct-freq-roll-out*
Nursery	50	100k	*direct-freq-roll-out*
TicTacToe	10	100k	*direct-freq-roll-out*
Yeast	20	100k	*large-freq-roll-out* ($jumpLength = 30$)

Table 11: The default parameters for each dataset.



(a) Runtime: BreastCanter (b) Avg. quality: Emotions (c) Descr length: Mushroom

Select Policies: (1) DFS-UCT with LO

(2) UCB1 (3) UCB1 with LO (4) UCB1 with PU

(5) SP-MCTS (6) SP-MCTS with LO (7) SP-MCTS with PU

(8) UCB1-Tuned (9) UCB1-Tuned with LO (10) UCB1-Tuned with PU

(11) UCT (12) UCT with LO (13) UCT with PU

Figure 14: Impact of the SELECT strategy.

the *BreastCancer* and *Emotions* datasets. LO leads to a worse quality in the result set, whereas PU seems to be more efficient.

The use of these different UCBs has no impact on the description length of the subgroups within the result set. For some datasets, the *permutation unification* leads to longer descriptions (see for instance Figure 14(c)).

4.4.4 The EXPAND method

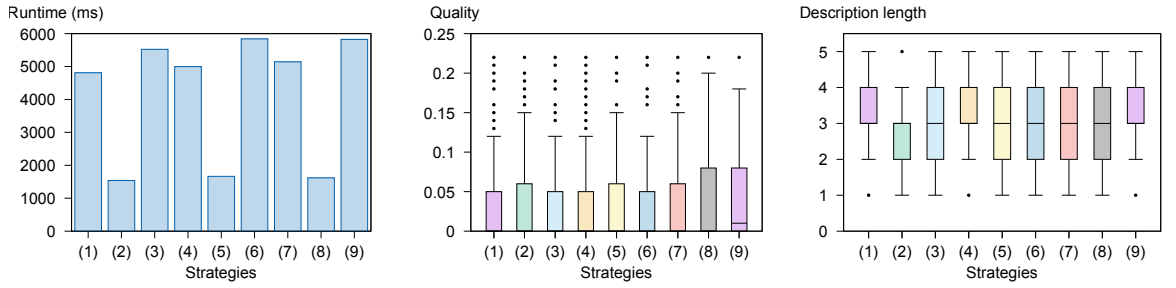
Considering the EXPAND method, we introduced three different refinement operators, namely *direct-expand*, *gen-expand* and *label-expand*, and we presented two methods, namely LO and PU, to take into account that several nodes in the search tree are exactly the same. The several strategies are given in Figure 15(bottom). Let us consider the leverage on the runtime of these strategies in Figure 15(a). Once again, using LO implies a decrease of the runtime. Conversely, PU requires more time to run. There is very little difference in the runtime when varying the refinement operator: *direct-expand* is the faster one, and *label-expand* is more time consuming.

Considering the quality of the result set varying the expand strategies, we can assume that the impact differs w.r.t. the dataset (see Figure 15(b)). Surprisingly, LO improves the quality of the result set for some datasets (e.g., the Iris dataset in Figure 15(b)). This contradicts what we observe in the Emotions dataset in Section 4.4.3. More generally, the results using *label-expand* are better than other ones in most of the datasets.

The description length of the extracted subgroups are quite constant when varying the EXPAND strategies (see Figure 15(c)). With LO, the description lengths are slightly smaller than with other strategies.

4.4.5 The ROLLOUT method

For the ROLLOUT step, we derived several strategies that combine the path policy and the reward aggregation policy in Table 12. Clearly, the experiments show that the runs using the direct re-



(a) Runtime: Nursery

(b) Avg. quality: Iris (c) Descr length: Mushroom

(1) *direct-expand* (2) *direct-expand* with LO (3) *direct-expand* with PU

(4) *gen-expand* (5) *gen-expand* with LO (6) *gen-expand* with PU

(7) *label-expand* (8) *label-expand* with LO (9) *label-expand* with PU

Figure 15: Impact of the EXPAND strategy.

finement operator (*naive-roll-out* and *direct-freq-roll-out*) are time consuming (see Figure 16(a)). In the BreastCancer data, the runtime is twice longer with the direct refinement operator than with the *large-freq-roll-out* path policy. In other datasets (e.g., Ionosphere or Yeast), the runtime is even more than 3 minutes (if the run lasts more than 3 minutes to perform the number of iterations, the run is ignored). Besides, it is clear that the *random-reward* aggregation policy is less time consuming than other strategies. Indeed, with *random-reward*, the measure of only one subgroup within the path is computed, thus it is faster.

Figure 16(b) is about the quality of the result set. The *naive-roll-out* and *direct-freq-roll-out* path policies lead to the worst results. Besides, the quality of the result set decreases with the *random-reward* reward aggregation policy in other datasets (e.g., Emotions). Basically, these strategies evaluate only random nodes and thus they are not able to identify the promising parts of the search space. Finally, there are not large differences between other strategies.

As it can be seen in Figure 16(c), the description length of the subgroups is not really impacted by the strategies of the ROLLOUT method. The results of the *random-reward* reward aggregation policy are still different from other strategies: The description length is smaller for the Mushroom dataset. Using *large-freq-roll-out* with $jumpLength = 100$ leads to smaller descriptions for the Mushroom dataset. Finally, the description length is however not really influenced by the ROLLOUT strategies.

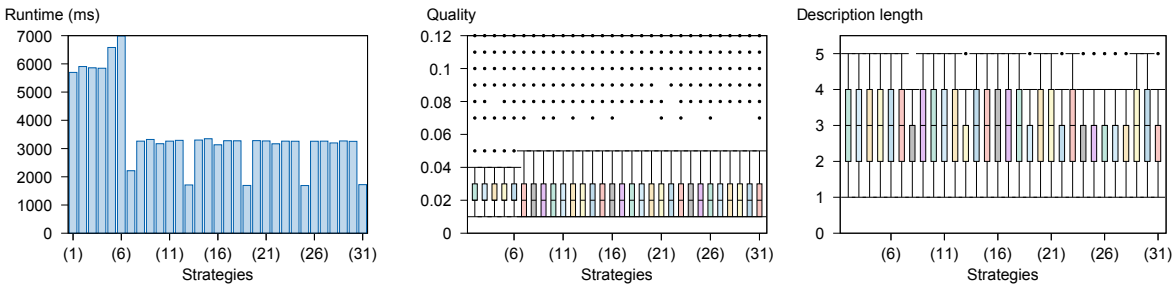
4.4.6 The MEMORY method

We derived six strategies for the MEMORY step given in Figure 17(bottom). Obviously, the *all-memory* policy is slower than other strategies because all the nodes within the path of the simulation have to be stored (see Figure 17(a)). Conversely, the *no-memory* policy is the fastest strategy. The runtimes of the *top-k-memory* policies are almost the same.

Figure 17(b) shows that the quality of the result set is impacted by the choice of the memory policies. We can observe that the *no-memory* is clearly worse than other strategies. Indeed, in the Emotion dataset, the best subgroups are located deeper in the search space, thus, if the solutions encountered during the simulation are not stored it is difficult to find them just be considering

Strategy	Path Policy	Reward Aggregation Policy
(1)	*naive-roll-out* ($pathLength = 20$)	*terminal-reward*
(2)	*direct-freq-roll-out*	*max-reward*
(3)	*direct-freq-roll-out*	*mean-reward*
(4)	*direct-freq-roll-out*	*top-2-mean-reward*
(5)	*direct-freq-roll-out*	*top-5-mean-reward*
(6)	*direct-freq-roll-out*	*top-10-mean-reward*
(7)	*direct-freq-roll-out*	*random-reward*
(8)	*large-freq-roll-out* ($jumpLength = 10$)	*max-reward*
(9)	*large-freq-roll-out* ($jumpLength = 10$)	*mean-reward*
(10)	*large-freq-roll-out* ($jumpLength = 10$)	*top-2-mean-reward*
(11)	*large-freq-roll-out* ($jumpLength = 10$)	*top-5-mean-reward*
(12)	*large-freq-roll-out* ($jumpLength = 10$)	*top-10-mean-reward*
(13)	*large-freq-roll-out* ($jumpLength = 10$)	*random-reward*
(14)	*large-freq-roll-out* ($jumpLength = 20$)	*max-reward*
(15)	*large-freq-roll-out* ($jumpLength = 20$)	*mean-reward*
(16)	*large-freq-roll-out* ($jumpLength = 20$)	*top-2-mean-reward*
(17)	*large-freq-roll-out* ($jumpLength = 20$)	*top-5-mean-reward*
(18)	*large-freq-roll-out* ($jumpLength = 20$)	*top-10-mean-reward*
(19)	*large-freq-roll-out* ($jumpLength = 20$)	*random-reward*
(20)	*large-freq-roll-out* ($jumpLength = 50$)	*max-reward*
(21)	*large-freq-roll-out* ($jumpLength = 50$)	*mean-reward*
(22)	*large-freq-roll-out* ($jumpLength = 50$)	*top-2-mean-reward*
(23)	*large-freq-roll-out* ($jumpLength = 50$)	*top-5-mean-reward*
(24)	*large-freq-roll-out* ($jumpLength = 50$)	*top-10-mean-reward*
(25)	*large-freq-roll-out* ($jumpLength = 50$)	*random-reward*
(26)	*large-freq-roll-out* ($jumpLength = 100$)	*max-reward*
(27)	*large-freq-roll-out* ($jumpLength = 100$)	*mean-reward*
(28)	*large-freq-roll-out* ($jumpLength = 100$)	*top-2-mean-reward*
(29)	*large-freq-roll-out* ($jumpLength = 100$)	*top-5-mean-reward*
(30)	*large-freq-roll-out* ($jumpLength = 100$)	*top-10-mean-reward*
(31)	*large-freq-roll-out* ($jumpLength = 100$)	*random-reward*

Table 12: The list of strategies used to experiment with the ROLLOUT method.

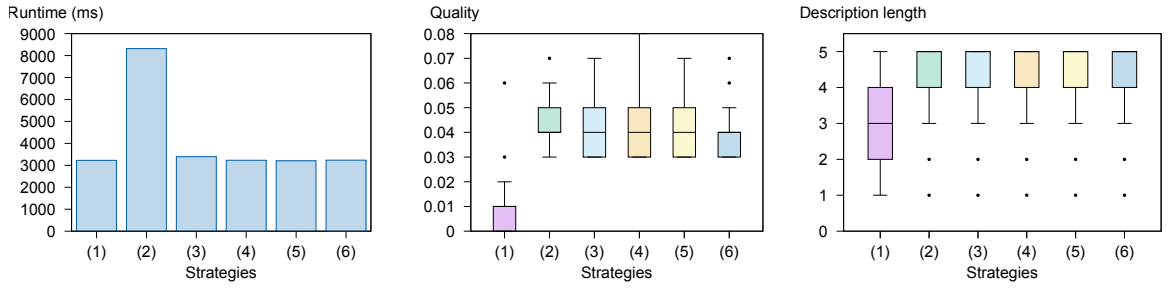


(a) Runtime: BreastCancer (b) Avg. quality: Mushroom (c) Descr length: Mushroom

Figure 16: Impact of the ROLL-OUT strategy.

the subgroups that are expanded in the search tree. Surprisingly, the *all-memory* policy does not lead to better results. In fact the path generated during a simulation contains a lot of redundant subgroups: Storing all these nodes is not required to improve the quality of the result set. Only few subgroups within the path are related to different local optima.

As expected in Figure 17(c), the descriptions of the subgroups obtained with the *no-memory* policy are smaller than those of other strategies. Indeed, with the *no-memory* policy, the result sets contains only subgroups that are expanded in the search tree, in other words, the subgroups obtained thanks to the EXPAND method.



(a) Runtime: Ionosphere (b) Avg. quality: Emotions (c) Descr. length: BreastCancer

(1) *no-memory* (2) *all-memory* (3) *top-1-memory*
(4) *top-2-memory* (5) *top-5-memory* (6) *top-10-memory*

Figure 17: Impact of the MEMORY strategy.

4.4.7 The UPDATE method

Figure 18(bottom) presents the different strategies we use to implement the UPDATE method. The goal of this step is to back-propagate the reward obtained by the simulation to the parent nodes. The runtime of these strategies are comparable (see Figure 18(a)). However, we notice that the *top-k-mean-update* policy is a little more time consuming. Indeed, we have to maintain a list for each node within the built tree that stores the top-k best rewards obtained so far.

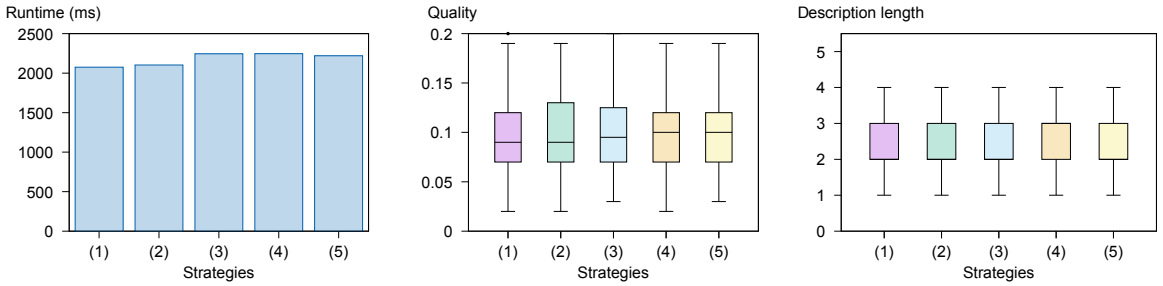
Figure 18(b) shows the quality of the result set when varying the UPDATE policies. For most of datasets, since the proportion of local optima is very low within the search space, the *max-update* is more efficient than the *mean-update*. Indeed, using the *max-update* enables to keep in mind that there is an interesting pattern that is reachable from a node. However, Figure 18(b) presents the opposite phenomena: The *mean-update* policy leads to better result. In fact, there are a lot of local optima in the Ionosphere dataset, thus the *mean-update* is designed to find the areas with lots of interesting solutions. Moreover, using the *top-k-mean-update* leads to the *mean-update* when k increases.

The description length of the subgroups in the result set are comparable when varying the policies of the UPDATE method (see Figure 18(c)). Indeed, the aim of the UPDATE step is just to back-propagate the reward obtained during the simulation to the nodes of the built tree to guide the exploration for the following iterations. This does not have a significant influence on the length of the description of the subgroups.

4.4.8 The number of iterations

We study the impact of different computational budgets allocated to MCTS4DM, that is, the maximum number of iterations the algorithm can perform. As depicted in Figure 19(a), the runtime is linear with the number of iterations. Note that the x-axis is not linear w.r.t. the number of iterations, please refer to the bottom of Figure 19 to know the different values of the number of iterations.

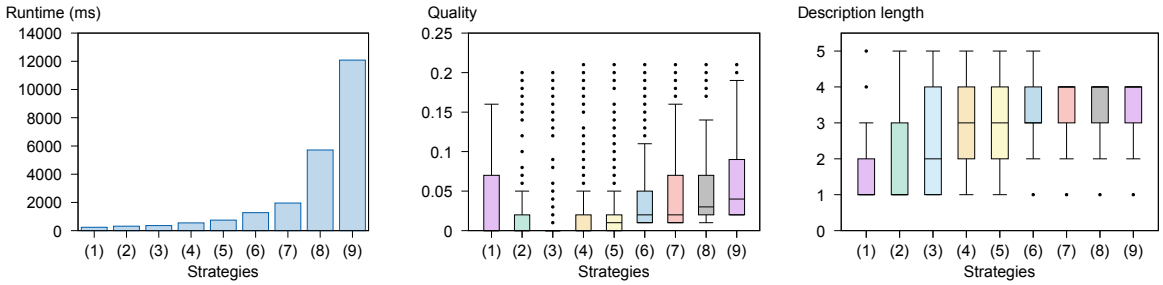
Moreover, as expected, the more iterations, the better the quality of the result set. Figure 19(b) shows that a larger computational budget leads to a better quality of the result set, but, obviously, it requires more time. Thus, with this instant mining exploration method, the user can have some results anytime. Note that for the *BreastCancer* dataset, the quality decreases from 10 to 100 iterations: This is due to the fact that with 10 iterations there are less subgroups extracted (12 subgroups) than with 100 iterations (40 subgroups), and the mean quality of the result set



(a) Runtime: TicTacToe (b) Avg. quality: Ionosphere (c) Descr length: Iris

(1) *max-update* (2) *mean-update* (3) *top-2-mean-update*
(4) *top-5-mean-update* (5) *top-10-mean-update*

Figure 18: Impact of the UPDATE strategy.



(a) Runtime: Cal500 (b) Avg. quality: BreastCancer (c) Descr length: Nursery

(strategy)#iterations: (1)10 (2)50 (3)100 (4)500 (5)1K (6)5K (7)10K (8)50K (9)100K

Figure 19: Impact of the maximal budget (number of iterations) .

with 100 iterations contains also subgroups with lower quality measures.

When the number of iterations increases, it leads to a result set containing subgroups with longer description lengths. Figure 19(c) depicts this behavior: The more the iterations, the longer the description length. Indeed, usually, the better the subgroups, the longer the description length. Note that in all the experiments, we limit the description length to at most 5 restrictions to remain interpretable. It is a popular constraint in SD.

4.4.9 The completeness of the result set

In this section we are interested in assessing the completeness of the result set of MCTS4DM. For that, we experiment with artificial data to get the ground truth, i.e., to know the percentage of local optima (hidden in the data) the algorithm is able to extract. We use the default parameter for MCTS4DM, and we select the *gen-expand* with the *top-10* memory policy. These parameters have been chosen empirically given the results obtained from benchmark datasets in the previous experiments. We set by default a very low minimal support of 10, that is 0.02%. The ability to retrieve patterns is measured with:

Definition 21 (Evaluation measure)**.** *Let $\mathcal{H}$ be the set of hidden patterns, and $\mathcal{F}$ the set of patterns found by an MCTS mining algorithm, the quality of the found collection is given by:*

$$qual(\mathcal{H}, \mathcal{F}) = avg_{\forall h \in \mathcal{H}}(max_{\forall f \in \mathcal{F}}(Jaccard(supp(h), supp(f))))$$

that is, the average of the quality of each hidden pattern, which is the best Jaccard coefficient with a found pattern. We thus measure the completeness. This measure is pessimistic in the sense that it takes its maximum value 1 *if and only if all patterns are completely retrieved.*

The results of our experiments are given in Figure 20. The impact of the noise is the following: It directly reduces the support of a hidden pattern and thus it requires low minimal supports to find the hidden patterns with success. With $minSupp = 50$ and no noise, pattern are discovered, however, with noise, the support of a pattern is lower that 50 and thus cannot be found. With $minSupp = 10$, the search is more tolerant to noise. Note on the Figures 20(a)-(b) that when two lines exactly overlaps, it means that the search space of frequent patterns was fully explored (the tree is fully extended in these cases). To ensure experiment feasibility, we had to reduce the number of hidden objects in the dataset.

With $out\_factor = 1$, each pattern appears equally often. Varying this parameter allows to hide patterns with different $WRacc$ measures. With 10K iterations, most of the patterns are retrieved, with 100K iterations all of them are found.

Varying the number of hidden pattern highlights the same results: The UCB allows to drive the search towards interesting parts (exploitation) but also rarely visited parts (exploration) of the search space. Hiding more patterns and retrieving all of them may require more iterations in the general case.

Patterns with a high relative support shall be easier to be retrieved as a simulation has more chance to discover them, even partially. We observe that patterns with small support can be retrieved in Figure 20(e) as the minimal support is set to 10 and the support of patterns is higher than 20. No pattern can be found if the $minSupp$ is higher than the support of the hidden patterns.

The number of objects directly influences the computation of the support of each node: Each node stores a *projected database* that lists which objects belong to the current pattern. The memory required for our MCTS implementation let show a linear complexity w.r.t. the number of iterations. This practical complexity can be higher depending on the chosen memory policy (e.g., in these experiments, the top-10 memory policy was chosen). The time needed to compute the support of a pattern is higher for larger datasets, but it does not change the number of iterations required to find a good result. The memory usage is discussed in Section 4.5.

The number of attributes and the size of attribute domains directly determine the branching factor of the exploration tree. It takes thus more iterations to fully expand a node and to discover all local optima. Here again, all patterns are well discovered but larger datasets require more iterations (figures not shown but similar).

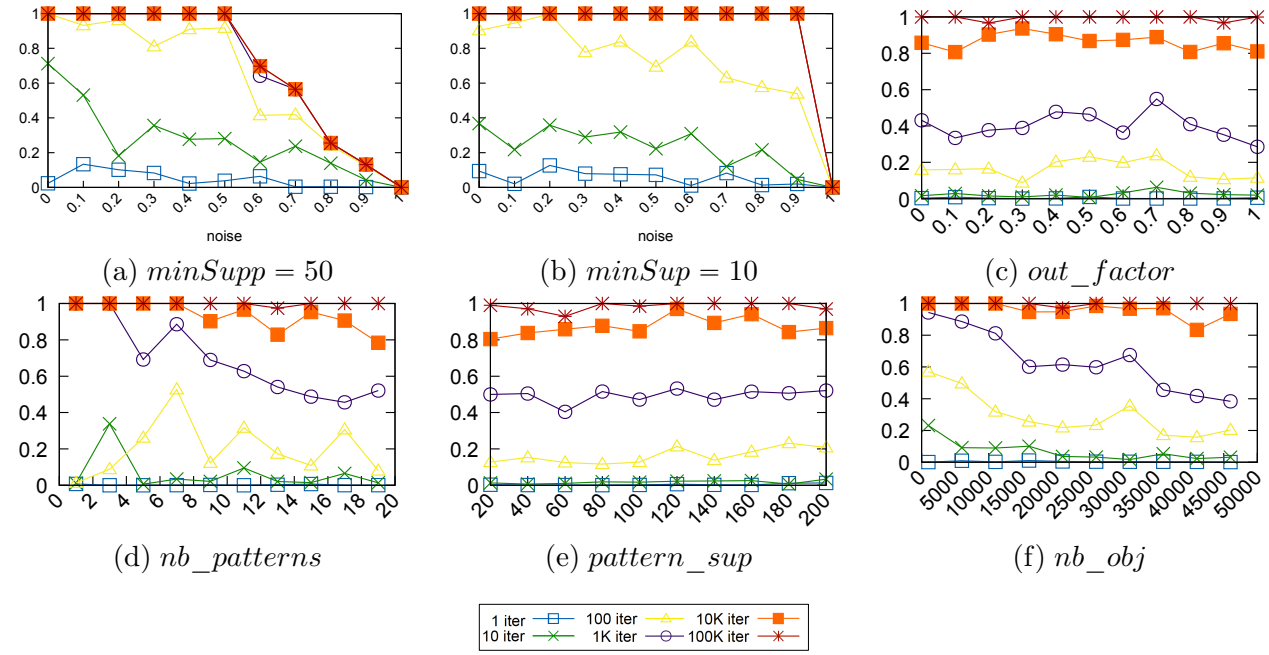
4.5 Comparison with existing methods

Let us consider the following existing implementations: SD-MAP*, the most efficient exhaustive search algorithm available to date [12] and its implementation in *Vikamine* [10] ; and CORTANA[8] which incorporates different versions of beam-search (standard one, ROC-search, cover-based search [112]). We did not compared to the beam search of DSSD [139], as a similar implementation is available in Cortana. We also did not compare to genetic algorithms and sampling method yet as we were unable to get the implementations from the authors[9].

In what follows, we compare these implementations with MCTS4DM given a choice of policies and a maximal budget in terms of iterations for (i) finding all hidden patterns in artificial data;

[8] `http://datamining.liacs.nl/cortana.html`

[9] However, as part of our revision in the journal *Data Mining and Knowledge Discovery*, it will be done soon.



(a) $minSupp = 50$ (b) $minSup = 10$ (c) out_factor

(d) $nb\_patterns$ (e) $pattern\_sup$ (f) nb_obj

Figure 20: $qual(\mathcal{H}, \mathcal{F})$ (Y-axis) for different datasets (X-axis)

(ii) finding the best pattern in benchmark data; and (iii) comparing the result set on a large real life dataset.

4.5.1 Studying extraction completeness in artificial data

We compare MCTS4DM and the default beam-search for their ability to retrieve hidden patterns in artificial data. Note that it is not necessary to compare with *SD-MAP** since it is a exhaustive search: The completeness of the result set is maximal. For that matter, we reuse our artificial data generator with the following default parameters: $noise\_rate = 0.2$, $out\_factor = 0.15$, $nb\_patterns = 25$ $pattern\_sup = 100$, $nb\_obj = 50000$, $nb\_attr = 25$, $domain\_size = 1000$. Again, we measure the quality of an extraction with Definition 21. Results are reported in Figure 21 when varying the out_factor, the number and the support of hidden patterns. It follows that MCTS4DM requires less than $5,000$ iterations to fully retrieve all patterns. Surprisingly, 100 iterations are enough to have similar results with those of beam-search. This is due to the fact that we generated datasets with a large number of attributes and domain sizes (the branching factor): The greedy choices of beam search fail at exploring well the search space. We obtained similar results with the other beam search strategies.

4.5.2 Finding the best pattern in benchmark datasets.

This experiment aims at comparing the best solution found with MCTS4DM after a given number of iterations, with the several versions of beam search. We also want to compare with SD-Map. As SD-Map performs an exhaustive search, it finds the best solution. Table 13(top) summarizes the results (note that the result of MCTS4DM are summed up from experiment reports introduced in the previous section).

For SD-Map, we use the following parameters: The WRAcc measure is used; the description

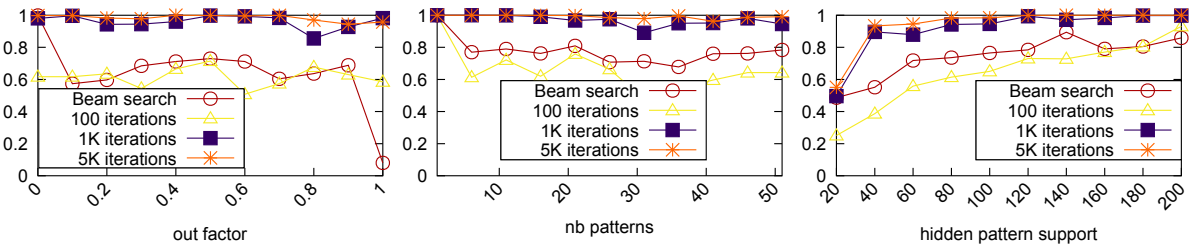


Figure 21: Comparing the diverstity after a beam search (Cortana) and MCTS4DM

length is fixed to 5; the minimum support is fixed in the same way than presented in Table 11 and the maximum size of the result set is 50. Results are given in Table 13(bottom). Surprisingly, the results extracted by SD-Map are not optimal for some datasets. Indeed, SD-Map uses a discretization method to handle numerical attributes. Thus for numerical dataset, the result set obtained with SD-Map may be not optimal and the runtime is biased since it is not exhaustive. On the *BreastCancer* dataset, MCTS4DM extracts a subgroup with a quality measure of 0.21 in 0.185 seconds, whereas the best quality measure with SD-Map is 0.184 and it took 0.7 seconds to find it. Besides, for the TicTacToe dataset, both approaches are able to extract the same best solution, but MCTS4DM is faster than SD-Map (0.173s vs 0.85s). However, for some datasets, e.g.Cal500, SD-Map is more efficient than MCTS4DM: It finds the best subgroup with a quality measure of 0.029 in 1.05 seconds whereas MCTS4DM does not extract it even after 100k iterations in 12.082 seconds (the best solution found so far is 0.027).

To compare with beam-search implementations, CORTANA can be configured with several parameters, such as the the quality measure, the time budget or the minimum support threshold. CORTANA embeds several beam search strategies: The traditional beam search, the ROC-Search [112] and the cover-based beam search (it used the similarity of the supports to choose the subgroups to add into the next beam). We use the following parameters:

- WRAcc measure is used.
- The refinement depth is fixed to 5. Note that the refinement depth is not exactly the description length. Indeed, the refinement depth of $[Attribute_1 < 5] \wedge [Attribute_1 < 4]$ is 2 but its description length is 1.
- The minimum support is fixed in the same way than presented in Table 11.
- The maximum size of the result set is 50.
- The time budget is set to 3 minutes.
- The numeric strategy is set to *best*.
- 1 thread is used since MCTS4DM is not a parallelized implementation yet.

Table 13(middle) gathers the results: The quality of the best subgroup is the same for most of the datasets, but the runtime to find them is much faster with MCTS4DM than with the strategies implemented in CORTANA. For few datasets, MCTS4DM has a lower quality measure because it needs more iterations to find these best solutions. On the Yeast dataset, none of these three strategies are able to find a solution without exceeding the time budget of 3 minutes, MCTS4DM does in less than 1 minute. MCTS4DM extracts the best solution with few iterations for many

MCTS4DM	1K iterations		50K iterations		100K iterations	
	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$
BreastCancer	**0.185**	**0.210**	3.254	**0.210**	6.562	**0.210**
Cal500	0.746	0.025	5.717	0.026	12.082	0.027
Emotions	0.770	0.054	7.090	0.068	14.024	0.069
Ionosphere	0.354	0.188	5.688	0.196	10.847	0.198
Iris	**0.105**	**0.222**	2.941	**0.222**	36.302	**0.222**
Mushroom	1.087	0.118	8.141	0.118	21.299	0.118
Nursery	1.334	0.076	5.653	0.076	5.701	0.076
TicTacToe	**0.173**	**0.069**	2.446	**0.069**	2.364	**0.069**
Yeast	4.776	0.027	26.976	0.032	49.785	0.032

Cortana	Beam search		ROC-Search		Cover-based Search	
	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$
BreastCancer	1.334	0.208	4.318	0.207	4.330	0.208
Cal500	**21.609**	**0.044**	>180	-	35.576	**0.044**
Emotions	**30.476**	**0.117**	>180	-	67.534	**0.117**
Ionosphere	15.482	0.202	4.618	0.201	**26.993**	**0.203**
Iris	1.335	**0.222**	1.664	**0.222**	1.996	**0.222**
Mushroom	1.591	0.173	10;512	0.173	5.554	0.173
Nursery	10.667	**0.145**	**4.219**	**0.145**	69.146	**0.145**
TicTacToe	1.335	**0.069**	1.340	**0.069**	1.364	**0.069**
Yeast	>180	-	>180	-	>180	-

	SD-Map	
	t(s)	$max(\varphi)$
BreastCancer	0.7	0.184
Cal500	1.05	0.029
Emotions	11.45	0.075
Ionosphere	0.97	0.069
Iris	0.73	0.164
Mushroom	**2.65**	**0.194**
Nursery	30.4	**0.145**
TicTacToe	0.85	**0.069**
Yeast	**47.37**	**0.055**

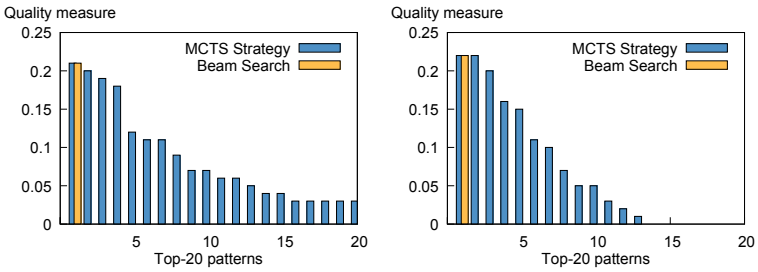
Table 13: The runtime and the maximum of the quality measure with SD-Map.

datasets. When the local optima are deep in the search space, it needs more iterations to find them. The result set of the beam search strategies contains only few local optima among a lot of redundant patterns, whereas MCTS4DM extracts more local optima (but still a lot of redundant patterns stored in the tree). This is illustrated in Figure 22: After applying the redundancy post-processing (that is, removing patterns that are too similar to other ones), it remains only one pattern with beam search in *BreastCancer* and *Iris* while MCTS4DM returns between 15 and 20 patterns.

MCTS may require a lot of iterations to find the best pattern (that maximize φ) and thus it needs for consequent memory to store the search tree (this is however nowadays a subdued problem as memory gets cheaper and cheaper). Due to its greedy nature, beam search may skip good solutions. When an exhaustive search is not able to complete quickly (see Section 4.5.3, since benchmark datasets are not too large), MCTS4DM outputs a result anytime that improves with time.

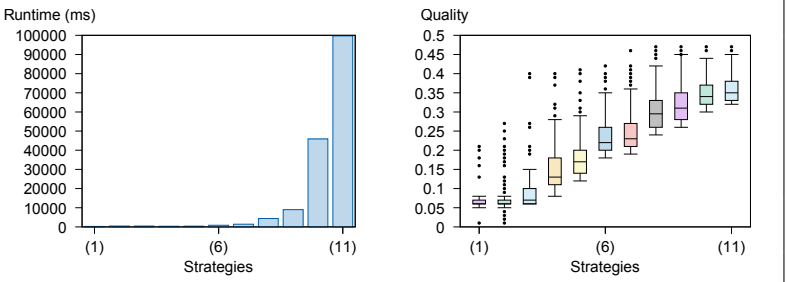
4.5.3 Studying the efficiency of MCTS4DM on a large dataset

Using MCTS for pattern mining is even more efficient when dealing with a real life application. Basically, the real life datasets are more complex and larger than benchmark datasets. We experimented with the olfaction dataset to understand the structure-odors relationships. This dataset contains 1,689 molecules described by 82 physico-chemical properties (e.g., the molecular weight,



(i) *BreastCancer* (ii) *Iris*

Figure 22: The top-20 subgroups after filtering out the redundant subgroups for both beam search and MCTS4DM.



(i) Runtime (ii) Quality measure

Figure 23: The runtime and the boxplots of the quality of the result set on the olfaction dataset with MCTS4DM when varying the number of iterations.

the number of carbon atoms, etc.) and associated to odors (e.g., strawberry, woody, etc.). Our goal is to extract subgroups that are characteristic of an odor or a subset of odors. In this experiment, we are only interested in the *Musk* odor, and we use the F1-score.

Comparison with the SD-Map algorithm. In the previous batch of experiments, all the dataset have been handled by the exhaustive algorithm *SD-Map*. Basically, the benchmark datasets are quite small, and the necessity of using heuristic searches is not perceptible. However, note that *SD-Map* uses a discretization method to handle numerical attributes that distorts the results: The run are faster but the result is not optimal. Experimenting SD-MAP on the olfaction dataset is not feasible in less than 3 minutes. The run lasts 477.8 seconds and it leads to the best quality measure of *F1-Score* = 0.45. Conversely, with MCTS4DM, we are able to process up to 1 million of iterations in 99, 7 seconds (average over 5 runs) with the best quality measure found is *F1-Score* = 0.47. Figures 23 presents the results obtained with MCTS4DM on the olfaction dataset when varying the number of iterations.

4.6 Conclusion

Heuristic search of supervised patterns becomes mandatory with large datasets. However, classical heuristics lead to a weak completeness in the result set: Only few local optima are found. We advocate for the use of MCTS for pattern mining: An aheuristic exploration strategy leading to

"any-time" pattern mining that can be adapted with different measures and policies. The experiments show that MCTS provides a much better diversity in the result set than existing heuristic approaches. Interesting subgroups are found thanks to a reasonable amount of iterations and the quality of the result iteratively improves. MCTS is a powerful exploration strategy that can be applied to several, if not all, pattern mining problems that need to optimize a quality measure given a subset of objects.

In this chapter, we apply MCTS4DM with the $WRAcc$ measure. However, when facing the SOR problem, it is required to employ a more complex model to capture the interestingness of subgroups. For that, in the next chapter, we are interested in finding subgroups that are relevant for a subset of class labels, called target subspaces. It leads to an increase of the branching factor in the search space. We study how MCTS4DM can compute, in this setting, a diverse set of patterns that is also diverse w.r.t. the target subspaces.

Chapter 5

Exceptional Model Mining in multi-label data

5.1 Introduction

We are now interested in multi-label data where an object can take several class labels. For that matter, SD has been extended to a richer framework that handles more complicated target concepts, the so-called exceptional model mining approach (EMM) [104]. The idea is the following: A model is built over the labels from the objects in the subgroup and it is compared to the model of the whole dataset by means of a quality measure (Figure 3). The more different the model, the more exceptional and interesting is the subgroup. For example, van Leeuwen and Knobbe compare the label distribution with the Weighted Kullback Leibler divergence (WKL) [139] and Duivesteijn et al. compare conditional dependency relations between the targets with Bayesian networks [61]. There exists many other types of models (e.g., regression, classification, target association) and associated quality measures (see [59]).

The proposed EMM instances for multi-label data consider either the whole set of labels (e.g., compared with the WKL) or each label independently (e.g., SD with the WRAcc), or finally a unique and fixed label subset chosen *a priori* (e.g. SD with the WRAcc for a label subset only). However, in presence of multi-label data, there are many applications for which one is interested in subgroups that differ from the whole dataset only on subsets of labels, though the interesting subsets of labels are not known beforehand. Typically, descriptive rules that conclude on small label sets provide such sets. As an example, in our case study on SOR, we are interested in rules between physico-chemical properties of odorant molecules (e.g. molecular weight, atom count), the attributes, and perceived smells given by Humans (e.g., fruity, apple, wood) which are the labels. There is a crucial need to discover such rules for a better understanding of the olfactory percept. According to a recent study, some odors can be predicted [91], but there is a few knowledge that would explain why a molecule smells an odor or another. On average, each odorant molecule is depicted by 2.33 labels among the set of the 74 possible labels. Assessing a subgroup with regard to either the whole set of labels or each label independently cannot allow to characterize rules involving a subset of labels. For example, Figure 24 shows the label distribution of the best subgroup according to the WKL in our olfaction dataset: It fails at characterizing a small set of labels.

The only solution for applying SD/EMM to successfully extract subgroups characterized by multi-labels is to transform the data. Indeed, when facing a multi-label dataset, either for a supervised or an unsupervised task, the most popular approaches apply standard techniques

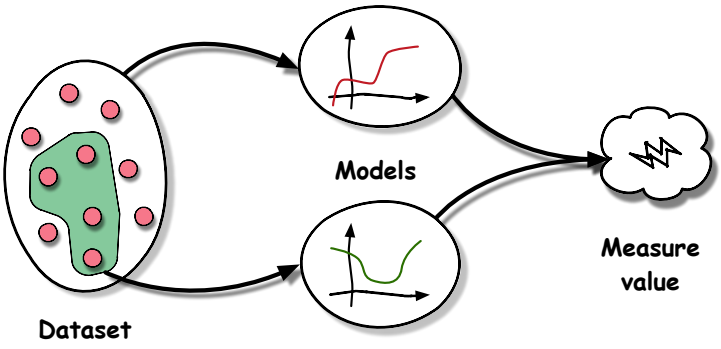


Figure 3: A typical EMM instance.

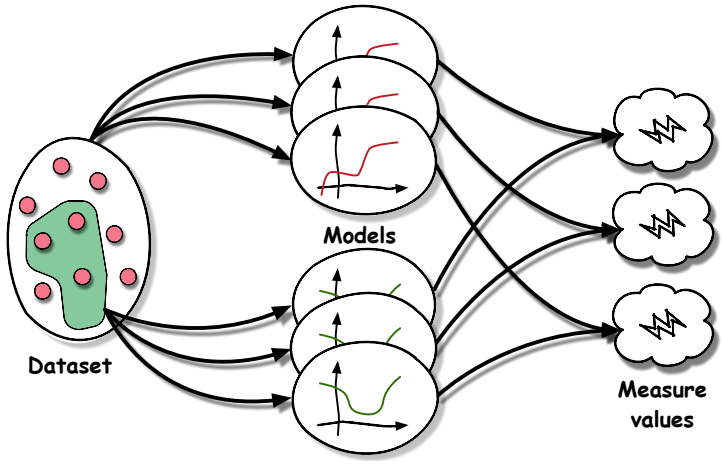


Figure 4: EMM considering target subspaces.

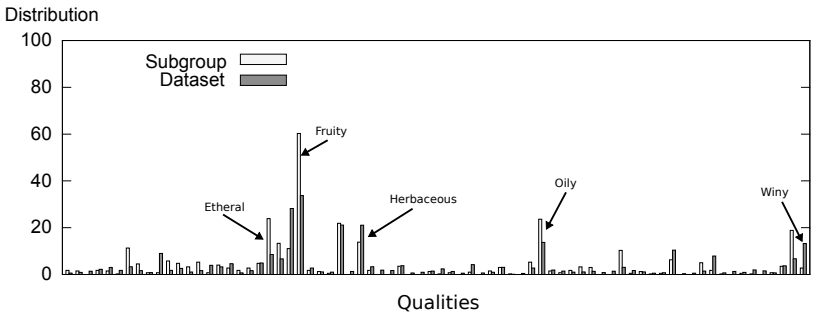


Figure 24: Label distributions of a subgroup output by EMM.

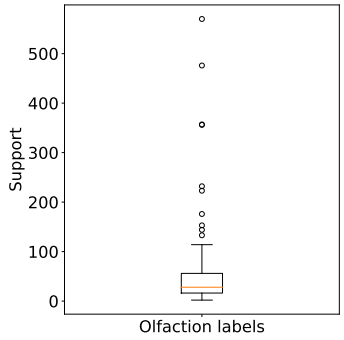


Figure 25: Olfaction data label distribution.

after a data transformation: Binary Relevance (BR) creates a dataset for each different label (objects are kept, target becomes binary) while Label Powerset (LP) creates a dataset for each set of labels that exists in the data set [134]. In our settings, BR looses label dependencies (e.g., fruity and apple), but applying LP before classical SD/EMM is totally relevant. It requires however to consider as many datasets as label subsets in the data, that is $1,800$ combinations in our olfaction dataset which could be restricted to 400 deemed as interesting by the neuroscientists

who created the dataset. This comes with an explosion of computational needs and an explosion of the number of rules returned to the expert. As explained hereafter, SD/EMM is nowadays shifting towards the heuristic search of a diverse set of subgroups that is large enough to cover interesting label sets, and small enough to be interpreted by an expert. As such, we properly formalize a novel EMM instance to fully take into account multi-label data and all label subsets called *target subspaces* (Figure 4). It leads to a more complex search space.

It has been now widely admitted that an exhaustive search is impossible in general, even with efficient pruning techniques [139, 59]: The subgroup search space is large and it is difficult to use the properties of the measures to prune the search space. Accordingly, heuristic approaches are used, mainly based on beam searches [108]. It as been studied as the *diverse subgroup set discovery* problem [139]: The extracted collection of k subgroups shall be of high quality and the less redundant as possible. Indeed, it often happens –if not always– that the best patterns differ slightly on the objects they cover (or their description in a dual way). The redundancy makes that very few local optima are discovered and the subgroup set is not diversified. A solution is to control each level of the beam search, which can be seen as a set of parallel hill-climbing searches [139, 59]: The subgroup search space is explored level-wise and each level is restricted to a set of diversified high quality patterns. The diversification is done as follows. Subgroups are sorted according to their quality: The best one is picked and all the next patterns that are too similar (according to similarity of their covers and a threshold) are removed. The first of the next patterns that is not similar is kept, and the process is iterated.

Besides, the problem of completeness of the result set with heuristic approaches has been studied in the previous chapter. The problem we tackle in this chapter concerns the diversity on the target space. This has not been studied yet to the best of our knowledge. Recall that we propose to enhance EMM by considering all target subspaces: It may happen that a subspace contains many subgroups of high quality and diversified in that subspace (the subgroups are very different). Therefore, we need to exploit promising subspaces while still exploring the others to cover many label combinations: This is what we call the *diversity in the target space*. To ensure the possibility of such a diversity, the search space of subgroups should include the label sets. We show that ensuring the possibility of this diversity forces to reconsider the subgroup search space: Bisets $(subgroup, label\_set)$ are explored and should be wisely expanded during the search. We experiment this with the standard beam-search, but also with our pattern sampling technique based on Monte Carlo Tree Search presented in the previous chapter.

Finally, as our goal is to find rules where the consequent is a label set, we also need to reconsider the subgroup quality measure. Generally, one is interested in rules with a high support and a few errors (maximizing the precision). The WRAcc is perfect for that as it expresses the difference between the precision of the subgroup w.r.t. the objects in the rest of the dataset, and it is weighted with the subgroup support. However, in many applications, the label distribution is highly skewed. This is the case of our application in olfaction as it can be observed on Figure 25: Some label subsets are over-represented, others are under-represented. The neuroscientists are also interested in subgroups that involve such under-represented label subsets. Another solution would be to consider both the precision and recall, thus the F_1 measure. However again, this would favor small groups for which it is easier to find descriptions that cover well the label subset. In the best settings, one would favor precision for highly represented label subsets, and both precision and recall for under-represented subsets. We thus propose the F_β measure adapted for highly skewed label distributions. Actually, the F_β measure generalizes the F_1 measure in a way that β expresses a trade-off between the precision and the recall: We dynamically adjust it during the search, given the target subspace (label subset) that is currently enumerated. This directly impacts a better target diversity in the result set.

To summarize, our main claims are manifold:

- We properly formalize a novel EMM instance to fully take into account multi-label data and all *target subspaces*.
- This comes with a more complex search space and it enforces new constraints for tractable heuristic searches: We show how to adapt the subgroup search space so that heuristic search, including MCTS on, can be applied efficiently.
- We introduce several pattern quality measures that are able to take into account the skewness of the label distribution and we dynamically consider the label distributions during the search to favor diversity on targets.
- We experiment with these approaches on several benchmark multi-label datasets (the experiments on the olfaction dataset are detailed in the next chapter). Our main result shows that MCTS with a relative F_β measure gives the best results in terms of computation time, quality and diversity, both on the description and the target spaces.

The rest of this chapter is organized as follows. The next section recalls some basics of SD/EMM. Section 5.3 formalizes EMM with different target spaces for multi-label rule discovery. Section 5.4 introduces the subgroup search space and the several quality measures that favor target diversity. Section 5.5 develops the different algorithms. Section 5.6 presents our experiments assessing the validity of our approach before a brief conclusion in Section 5.7.

5.2 Exceptional model mining and diverse subgroup set discovery

We override the definition of the label dataset with the Definition 7 as follows:

Definition 7 (Multi-label dataset)**.** *Let $\mathcal{O}$, $\mathcal{A}$ and $\mathcal{C}$ be respectively a set of objects, a set of attributes (either nominal or numerical), and a set of class labels. Each object is associated to a subset of class labels among $\mathcal{C}$ by the function $class : \mathcal{O} \mapsto 2^{\mathcal{C}}$ that maps the target labels to each object. We denote a multi-label dataset as $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$.*

In the EMM framework, with a multi-label dataset, the definition of a subgroup remains unchanged. We recall this definition:

Definition 6 (Subgroup)**.** *The description of a subgroup is given by $d = \langle f_1, \ldots, f_{|\mathcal{A}|} \rangle$ where each f_i is a restriction on the value domain of the attribute $a_i \in \mathcal{A}$. The description of a subgroup corresponds to the pattern in the pattern mining setting. A restriction is either a subset of a nominal attribute domain, or an interval contained in the domain of a numerical attribute. The description d covers a set of objects called the extent of the subgroup, denoted $ext(d) \subseteq \mathcal{O}$, and its support is the number of objects in its extent and is defined by $supp(d) = |ext(d)|$. Note that, in this thesis, we denote $|S|$ the cardinality of the set S. For simplicity, a subgroup is either given by its intent, i.e., its description d, or by its extent $ext(d)$.*

The aim of EMM is to find subgroups whose model over the class labels is significantly different from the model induced by the entire set of objects $\mathcal{O}$. Several models have been proposed in the state of the art (see Chapter 2 for more details) [59]. Quality measures have been introduced to compare the similarity between the two models (i.e., those of the subgroup and those of the

entire dataset), the better the quality measure, the less the similarity, the more interesting the subgroup [102, 2].

Since the search space of subgroups is too large, it is now widely accepted that an exhaustive search of subgroups, even with efficient pruning techniques, is not tractable for EMM. Heuristic methods are employed, such as beam search. However, it comes with the issue of redundancy of the subgroup set that is extracted. The question is: How to ensure a high diversity in the result set of a heuristic exploration of the search space. This point has been studied as the Diverse Subgroup Set Discovery [139]. The aim is to extract a diverse subgroup set that is as small as possible to be easily interpretable by the experts. For that, a similarity measure (e.g., Jaccard coefficient between the extent of two subgroups) is used to avoid the extraction of redundant subgroups within the result set. Thus, the problem definition of EMM given Problem 3 is extended with the problem of diverse subgroup set discovery as follows:

Problem 6 (Diverse Subgroup Set Discovery [139]). *Given a multi-label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a quality measure φ , a minimum support threshold minSupp, an integer k, a similarity measure sim, and a maximum similarity threshold Θ, DSSD aims at extracting the diverse set of top-k best frequent subgroups w.r.t. the quality measure φ in which there is no similar subgroups, i.e., for all subgroups s_1, s_2 in the result set, $sim(s_1, s_2) \leqslant \Theta$.*

5.3 Subgroup set discovery with diversity on target subspaces

By definition, within the EMM framework, the model induced by the subgroup (and those induced by the entire set of objects) is always built over *all* the class labels. Thus, each subset of objects (or subgroup) derives a unique model. However, a subgroup can be deemed interesting only for the model induced over a subset of class labels because it derives a model completely different from those of the entire set of objects just for this subset of class labels. However, this subset of class labels is unknown *a priori*. Thus, it is required to explore the label set space, called the *target subspaces*. This is one of our contribution: We design a new EMM instance to strive subgroups whose model induced over an unknown subset of class labels $L \subseteq \mathcal{C}$ is different from the model induced by the entire dataset over the same subset of class labels L. The process of this new EMM instance is given in Figure 4. The change relies on the construction of the model. There is no longer *one* but several models derived from the subgroup: One for each target subspace. We need to refine Definition 6 about subgroups for this new EMM instance.

Definition 22 (Subgroup in a target subspace). *Given a multi label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a subgroup, denoted $s = (d, L)$, is given by its description d and the subset of class labels $L \subseteq \mathcal{C}$ over which the model is built.*

We will use the term *subgroup* in the rest of this chapter though, as it will be always implied that a subgroup is considered in a target subspace. Following the studies in *Diverse Subgroup Set Discovery*, this new EMM instance that deals with multi-label data should also exhibit a great diversity in the target subspace. Although it may happen that a subspace contains many subgroups of high quality and diversified in that subspace (the subgroups are very different), we need to exploit promising target subspaces while still exploring the others to cover many label combinations: This is what we call the diversity in the target space.

Problem 7 (DSSD with diveristy on target subspaces). *Given a multi-label dataset $\mathcal{D}(\mathcal{O}, \mathcal{A}, \mathcal{C}, class)$, a quality measure φ, a minimum support threshold minSupp, an integer k, and a similarity measure sim, DSSD aims at extracting the diverse set of top-k best frequent subgroups w.r.t.*

the quality measure φ in which there is no similar subgroups w.r.t. sim and that covers as many target subspaces as possible.

We will still refer to this problem as DSSD, as it implies in the rest of this chapter that diversity is considered both on the description and target subspace.

5.4 Quality measures considering the target subspaces

Evaluating a subgroup $s = (d, L)$ is performed thanks to a quality measure φ that computes the difference between the model induced by $ext(d)$ over the target subspace L and those induced by $\mathcal{O}$ over L. Surveys help understanding how to choose the right measure [68]. One of the most widely used quality measure for multi-label data is the Weighted Kullback Leibler divergence (WKL) [139]. The WKL of a subgroup $s = (d, L)$ is given by:

$$WKL(d, L) = \frac{supp(d)}{|\mathcal{O}|} \sum_{l \in L} (p_d^l \log_2 \frac{p_d^l}{p_0^l})$$

This measure aims at assessing the deviation between two distributions, i.e., it does not consider only the presence of a label, but it also takes into account the under-representation of a label for a subgroup. Moreover, WKL assumes that the labels are independent: It does not consider the co-occurrences of the labels. This is a strong assumption that is not satisfied in most of the data. Besides, note that *WKL* is maximized with $L = \mathcal{C}$ since it is the sum over the labels in L of positive terms. Thus, *WKL* can not be used in the settings of Problem 7. Let us now study different measures that can be used for Problem 7.

5.4.1 *WRAcc* to evaluate the subgroups

The Weighted Relative Accuracy (*WRAcc*) is a well-known quality measure in EMM. Indeed, it allows to compare the proportion of a subset of labels in a subgroup with the proportion of this subset of labels in the entire dataset. For a subgroup $s = (d, L)$, it is given by:

$$WRAcc(d, L) = \frac{supp(d)}{|\mathcal{O}|} \times \left(p_d^L - p_0^L\right)$$

where $p_d^L = \frac{|\{o \in ext(d) | class(o) \subseteq L\}|}{supp(d)}$ (resp. $p_0^L = \frac{|\{o \in \mathcal{O} | class(o) \subseteq L\}|}{|\mathcal{O}|}$) is the proportion of objects in the subgroup s (resp. in the entire dataset) that are associated to all the labels in L. In other words, *WRAcc* is the difference between the precision of the rule $d \rightarrow L$ and those of rule $\langle\rangle \rightarrow L$: The model of a subgroup s is given by the precision of the subset of labels L in $ext(s)$. This difference is weighted by the relative size of the subgroup to avoid the extraction of small subgroups. Note that, we can also consider the Relative Accuracy measure (*RAcc*) that does not weight the difference:

$$RAcc(d, L) = p_d^L - p_0^L$$

However, in our case study, we are interested in both the precision and the recall of the subgroup. *WRAcc* or *RAcc* only foster on the precision of a given subgroup. We can notice that the weighted factor allows to take into account partially the recall by fostering on larger subgroups.

Table 8: The extension of the label dataset from Table 7 to the multi-label dataset version.

ID	a	b	c	$class(\cdot)$
1	150	21	11	$\{l_1,\ l_3\}$
2	128	29	9	$\{l_2\}$
3	136	24	10	$\{l_2,\ l_3\}$
4	152	23	11	$\{l_3\}$
5	151	27	12	$\{l_1,\ l_2\}$
6	142	27	10	$\{l_1,\ l_2\}$

Example 18. *Let us consider the multi-label dataset given in Table 8. For the description* $d = \langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$ *we can induce 7 different models, one for each subset of* $\mathcal{C}$*, namely* $\{l_1\}$*,* $\{l_2\}$*,* $\{l_3\}$*,* $\{l_1,\ l_2\}$*,* $\{l_1,\ l_3\}$*,* $\{l_2,\ l_3\}$ *and* $\{l_1,\ l_2,\ l_3\}$*. With the WRAcc measure:* $\varphi(d, \{l_1\}) = \frac{4}{6} \times (\frac{2}{4} - \frac{3}{6}) = 0$*,* $\varphi(d, \{l_2\}) = 0.22$*,* $\varphi(d, \{l_3\}) = -0.17$*,* $\varphi(d, \{l_1, l_2\}) = 0.11$*,* $\varphi(d, \{l_1, l_3\}) = -0.11$*,* $\varphi(d, \{l_2, l_3\}) = 0.06$ *and* $\varphi(d, \{l_1, l_2, l_3\}) = 0$*. Thus the best model induced by the description* d *is obtained for the subset of labels* $\{l_2\}$*.*

5.4.2 F_1 score to take into account both precision and recall

Given a subgroup $s = (d, L)$, the precision is defined by:

$$P(d, L) = \frac{|ext(d) \cap ext(L)|}{supp(d)}$$

and the recall is given by:

$$R(d, L) = \frac{|ext(d) \cap ext(L)|}{supp(L)}$$

The F_1 score considers both the precision and the recall of a subgroup $s = (d, L)$. The Relative F_1 (RF_1) is given by:

$$RF_1(d, L) = F_1(d, L) - F_1(\langle\rangle, L)$$

where $F_1(d, L) = 2 \times \frac{P(d,L) \times R(d,L)}{P(d,L) + R(d,L)}$. Indeed, objects are described by both attributes and class labels, so the $F1$ score quantifies both the precision and the recall of the extent of the description w.r.t. the extent of the class labels. Moreover, we can also consider the Weighted Relative F_1 (WRF_1), that uses the relative support size of the subgroup to weight RF_1:

$$WRF_1(d, L) = \frac{supp(d)}{|\mathcal{O}|} \times RF_1(d, L)$$

However, in most of the datasets, the distribution of class label is imbalanced. Some labels are associated to many objects in the dataset, and others are rarely used to label the data. Taking into account this setting is essential in this new EMM instance because the quality measure of the subgroups related to a frequent subset of class labels can be biased. F_1 does not equally evaluate subgroups related to over-represented subsets of labels and subgroups related to labels that are not over-represented. Empirically, we demonstrate that RF_1 is not effective to discover subgroups related to over-represented labels.

Example 19. *Again, let us consider the toy dataset in Table 8 and the subgroup with description* $d = \langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$*. With* $L = \{l_2\}$*, the model induced by* d *is* $F_1(d, \{l_2\}) = 1$*. The model induced by the entire dataset is* $F_1(\langle\rangle, \{l_2\}) = 0.8$*. Thus,* $\varphi(d, \{l_2\}) = RF_1(d, \{l_2\}) =$

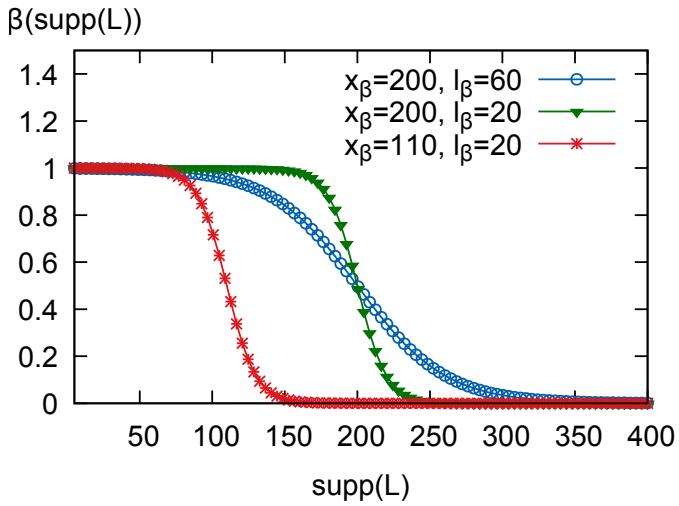


Figure 26: The curves of $\beta(supp(L))$.

$F_1(d, \{l_2\}) - F_1(\langle\rangle 0, \{l_2\}) = 0.2$. *With* $L = \{l_1, l_2\}$, *the models are* $F_1(d, \{l_1, l_2\}) = 0.66$ *and* $F_1(\langle\rangle, \{l_1, l_2\}) = 0.5$, *and thus* $RF_1(d, \{l_1, l_2\}) = 0.16$. *Using* WRF_1 *as quality measure, the result is* $WRF_1(d, \{l_2\}) = 4/6 \times RF_1(d, \{l_2\}) = 0.13$ *and* $WRF_1(d, \{l_1, l_2\}) = 4/6 \times RF_1(d, \{l_1, l_2\}) = 0.11$

5.4.3 An adaptive F_β-score for skewed label distributions

The *WRAcc* focuses, by definition, on the precision of labels and it promotes subgroups in populated target subspaces. The F_1 score makes a trade-off between precision and recall but we demonstrate empirically that this measure promotes subgroups covering few objects with low frequency label combinations. The trade-off lies with the so called F_β score which adds a parameter β that allows to tune the importance of the recall.

We propose β to be a function of the support of the considered target subspace L, so that the trade-off can be automatically adapted during the search, $\beta(supp(L))$. The greater $supp(L)$, the closer to zero β is: The precision is fostered in F_β for *over-represented* labels. Conversely, the lower $supp(L)$, the closer to one β is: F_β is equivalent to the traditional F_1 score (the harmonic mean of precision and recall) for *non over-represented* labels. Formally, given two positive real numbers x_β and l_β, we define the Relative F_β score (RF_β) as follows (see also Figure 26):

$$RF_\beta(d, L) = F_\beta(d, L) - F_\beta(\langle\rangle, L)$$

where $F_\beta(d, L)$ is defined as follows:

$$F_\beta(d, L) = (1 + \beta(supp(L))^2) \times \frac{P(d, L) \times R(d, L)}{(\beta(supp(L))^2 \times P(d, L)) + R(d, L)}$$

where $\beta(supp(L))$ is:

$$\beta(supp(L)) = 0.5 \times \left(1 + \tanh\left(\frac{x_\beta - supp(L)}{l_\beta}\right)\right)$$

Intuitively, for over-represented labels in the data, since it is difficult to find rules with high recall and precision, the experts prefer to foster the precision instead of the recall. They prefer

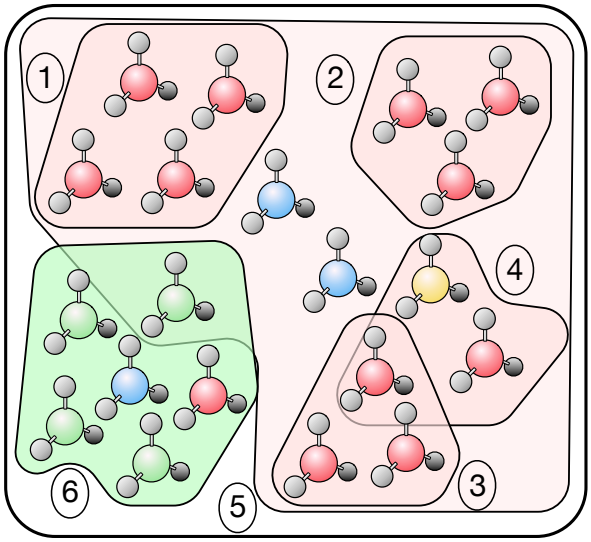


Figure 7: Necessity of an adaptive measure.

extracting several small subgroups with a high precision than a huge subgroup (d, L) with plenty of non-L objects. In Figure 7 the red molecules are over-represented in the dataset, but it is more interesting having the different subgroups 1, 2, 3 and 4 with high precision, rather than a single huge local subgroup (5) which precision is much lower. For molecules that are not over-represented, the measure considers both precision and recall: e.g., subgroup 6 is possible for the green molecules.

Similarly to WRF_1 we can define the Weighted Relative F_β score as follows:

$$WRF_\beta(d, L) = \frac{supp(d)}{|\mathcal{O}|} \times RF_\beta(d, L)$$

Example 20. *Let us take the example of the multi-label dataset in Table 8, with the description* $d = \langle [128 \leqslant a \leqslant 151], [23 \leqslant b \leqslant 29] \rangle$ *again. We consider the* F_β *measure to evaluate a model and* RF_β *to evaluate the subgroup. Since* $supp(l_1) = 3$, $supp(l_2) = 4$ *and* $supp(l_3) = 3$, *we set* $x_\beta = 3.3$ *and* $l_\beta = 0.5$. *The* β *value for the subset of labels* $\{l_2\}$ *is 0.06 since in this setting,* l_2 *is over-represented within the dataset. The model induced by* d *on* $L = \{l_2\}$ *is* $F_\beta(d, \{l_2\}) = 1$. *The model induced by the entire dataset is* $F_\beta(\langle\rangle, \{l_2\}) = 0.67$. *Thus,* $\varphi(d, \{l_2\}) = RF_\beta(d, \{l_2\}) = F_\beta(d, \{l_2\}) - F_\beta(\langle\rangle, \{l_2\}) = 0.33$. *With* WRF_β, *we have* $WRF_\beta(d, \{l_2\}) = \frac{4}{6} \times RF_\beta(d, \{l_2\}) = 0.22$ *For the subset of class labels* $\{l_1, l_2\}$, $\beta = 0.99$. *The models are* $F_\beta(d, \{l_1, l_2\}) = 0.66$ *and* $F_\beta(\langle\rangle, \{l_1, l_2\}) = 0.5$, *and thus* $\varphi(d, \{l_1, l_2\}) = 0.16$. *Using* WRF_β, *the quality measure of* s *is* $WRF_\beta(d, \{l_1, l_2\}) = \frac{4}{6} \times RF_\beta(d, \{l_1, l_2\}) = 0.11$. *Note that for* $L = \{l_1, l_2\}$, RF_β *and* WRF_β *are equivalent to* RF_1 *and* WRF_1 *since* L *is not over-represented in the data.*

5.5 Search space explorations

We present the search space of subgroups that needs to be traversed in all target subspaces. We briefly detail an algorithm for exhaustive search as a baseline for some of our experiments. We then present the two heuristic search techniques that we employ in our experiments.

5.5.1 Search space and exhaustive search

In standard SD and EMM, the search space of subgroups is given by the lattice of all possible descriptions $(D, \sqsubseteq)$ where $d_1 \sqsubseteq d_2$ means that subgroup d_1 is more general than d_2, or equivalently $ext(d_2) \subseteq ext(d_1)$. This lattice can be explored either in a depth-first (DFS) or in a breadth-first (BFS) search manner. During the traversal, the quality measure is computed for each subgroup. Contrary to the extent, the monotonic property does not necessary hold for the quality measure. In the end, a redundancy filter based on the similarity function *sim* and the maximum similarity threshold Θ is applied to output the top-k diverse subgroups.

In our case, we wish to evaluate a description on each of the target subspaces. We need to consider the following search space: $D \times 2^{\mathcal{C}}$, where $\mathcal{C}$ is the set of labels. Hence, a subgroup is always considered in a target subspace in which the quality measure can be computed. Thus, only slight modifications in existing algorithms are required. We override the specialization/generalization relation $\sqsubseteq$ as follows: $(d_1, L_1) \sqsubseteq (d_2, L_2) \iff ext(d_2) \subseteq ext(d_1) \wedge L_2 \subseteq L_1$

For that, we adapt the algorithm *CloseByOne* [100] from the Formal Concept Analysis [70] that can handle easily both nominal and numerical attributes [88]. Without entering into the details, it avoids to generate subgroups having exactly the same support and truly operates an exhaustive search. Indeed, we could have adapted the most efficient subgroup discovery algorithm, SDMap* [11], but it is not purely exhaustive as it operates greedy cutting of numerical attributes. Moreover, we focus on heuristic search and already shown that MCTS performs better than *SDMap** for large search space in the previous chapter.

5.5.2 Heuristic search with Beam-search

Beam search [108] is the most popular heuristic technique in SD/EMM. It has been originally adapted to consider the *diverse subgroup set discovery* problem by Leeuwen and Knobbe [139]. The subgroup search space is explored level-wise (BFS) and each level is restricted to a set of diversified high quality patterns. The diversification is done as follows. Subgroups are sorted according to their quality: The best is picked and all the next patterns that are too similar (bounded Jaccard coefficient between their support) are removed. The first of the next patterns that is not similar is kept, and the process is iterated.

Adapting beam search considering the search space $D \times 2^{\mathcal{C}}$ with diversity on the target subspaces is done as follows. It starts from the most general subgroup. Next levels are generated by specializing subgroups either by restricting an attribute or by extending the subset of class labels with a new label as long as the quality measure is improved. There are at most $|\mathcal{C}| + \sum_{a_i \in \mathcal{A}} |a_i|(|a_i| + 1)/2$ possibilities to specialize each subgroup: We can proceed up to $|\mathcal{C}|$ extensions of the subset of labels to characterize L and $|\mathcal{A}|$ extensions of the description for which we can build up $|a_i|(|a_i| + 1)/2$ possible intervals for numeric attributes. Note that in beam search, an attribute is refined at most once: We do not restrict an effective restriction anymore. We choose among those only a constant number of candidates to continue the exploration, i.e., the beam width: The $beamWidth$ best subgroups w.r.t. the quality measure. Removing the redundancy is done as in the traditional case, but each subspace is considered separately to avoid to favor few excellent target subspaces: The subgroups are split into groups according to their target subspace, and the diversity filter is operated on each of them.

5.5.3 Sampling patterns with Monte Carlo Tree Search

We propose to sample the subgroup search space $D \times 2^{\mathcal{C}}$ relying on Monte Carlo Tree Search [36]. Indeed, in the previous chapter, we showed that MCTS is able to handle very large search

space and outperforms beam search in terms of quality and completeness of the result set when considering basic subgroup discovery and WRAcc. We recall that MCTS is a budget based approach: The more time and memory allocated, the better the result.

As a brief reminder, MCTS iteratively draws a random subgroup description s_n from the expanded node s_{exp}, called s_0 in this case, following a path $s_0 \sqsubset s_1 \sqsubset s_2 \sqsubset ... \sqsubset s_n$. The best pattern quality measure found on the path is returned as a reward. s_0 is stored in a memory (the search tree) and the reward is back-propagated in the tree: Each node stores the number of times it was visited and the average quality measure obtained so far. The tree will drive the search for the next iterations (thanks to the upper confidence bound, a formula that expresses a trade-off between exploration and exploitation of the search space [95]). In other terms, a new node (s_0) will be expanded into the tree according to this trade-off.

More specifically, MCTS operates a fixed number of iterations where: (i) It *selects* the most urgent node, called the selected node and denoted s_{-1} in this case (s_{sel} in the previous chapter), in the tree according to the UCB; (ii) it *expands* this node by randomly selecting one of the direct description specialization $s_{-1} \sqsubset d_0$ (the expanded node also called s_{exp}), (iii) it *simulates* a random path $s_0 \sqsubset s_1 \sqsubset s_2 \sqsubset ... \sqsubset s_n$, and (iv) the best reward $\varphi(s_i)$ found on the path is used to *update* the tree. Each of these four steps can be achieved with many strategies. We use the best settings found for the WRAcc in basic subgroup discovery settings.

5.6 Experimental study

The subgroups we consider are equivalent to the so called *Multi-label and sparse label-independent rules* in [109]. In this article, the authors have defined a list of 8 types of rules in single and multi-label data. However, they focus on rules where only a single label can appear in the head of the rule and other label can appear in the body. They opt for a kind of binary relevance transformation (each label is learned separately) that is not suitable in our settings and thus is not included in our experimental evaluation. Experiments were carried out on an Intel(R) Core(TM) i7-7700HQ CPU 2.80 GHz machine with 8 GB RAM. All materials are available on `https://github.com/guillaume-bosc/EMM_FBeta`. After introducing the different datasets, we will answer to the following questions:

- How to tune x_β and l_β parameters of the F_β measure?
- Is the MCTS approach also able to cope with the (Weighted) Relative F_β?
- What is the best measure to ensure the diversity on the target subspaces with MCTS?
- Is that measure also the best for beam search? It will thus advocate that the measure is not dependent of the heuristic.
- Is the (weighted) relative F_β ranking the subgroups differently? It will thus empirically show that the other measures are not equivalent.
- How are precision and recall of the subgroup distributed w.r.t. target subspace frequency? We should observe that the precision-recall trade-off is respected in the result sets: Fostering on precision for over-represented label sets, and also on the recall for the others.

Dataset	$\|\mathcal{O}\|$	$\|\mathcal{A}\|$	*Domain*	$\|\mathcal{C}\|$	*cardinality*	*density*	*distinct*	*mean*	*median*
CAL500	502	68	numeric	174	26	0.15	502	75.14	39
emotions	593	72	numeric	6	1.8	0.31	27	184.67	170
enron	1702	1001	nominal	53	3.3	0.06	753	108.49	26
genbase	662	1186	nominal	27	1.2	0.04	32	30.70	17
yeast	2417	103	numeric	14	4.2	0.30	198	731.5	659
Olfaction	1689	82	numeric	74	2.882	0.04	1069	65.80	28

Table 14: Datasets used for the experiments.

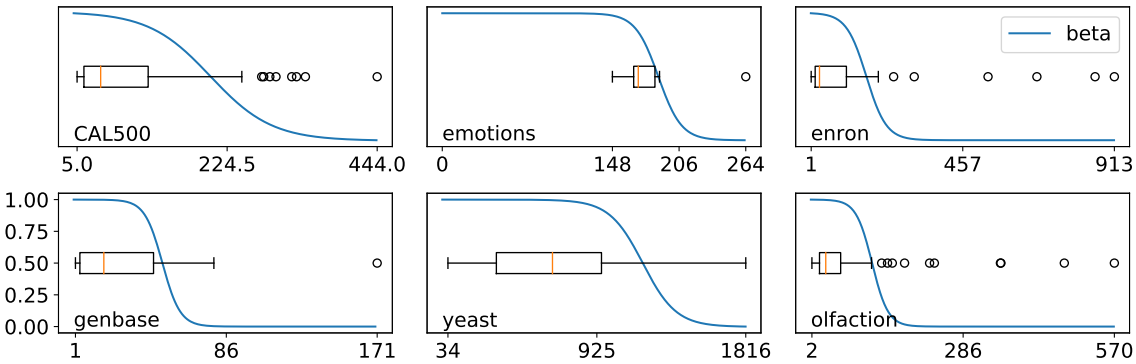


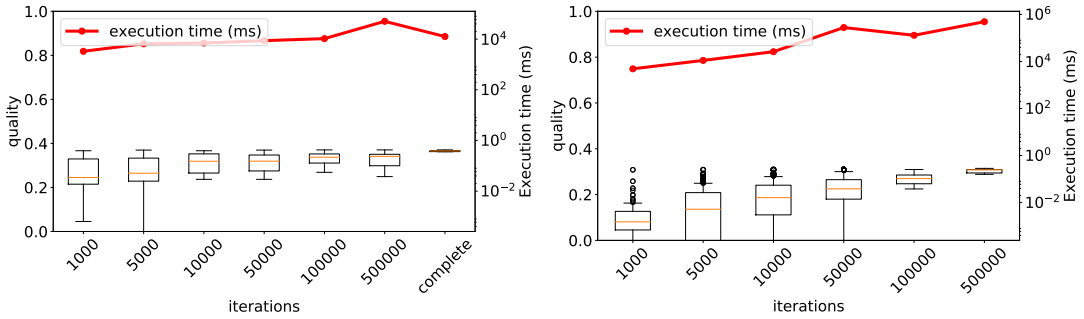
Figure 27: Label distribution in the different datasets of Table 14.

5.6.1 Data

We experiment our approach with a set of multi-label datasets from the well-known Mulan repository [135]. We also experiment with the olfactory dataset built by a neuroscientist (co-author of the present work) which is explained in the next chapter where we assess the interest of our approach in a real world scenario. Table 14 gives the properties of each dataset: The number of instances $|\mathcal{O}|$, the number of attributes $|\mathcal{A}|$, the domain of the attributes (nominal or numeric), the number of labels $|\mathcal{C}|$, average number of labels associated to an object (cardinality), its density (the cardinality divided by the number of labels) and the mean and median of the number of objects per label. Figure 27 finally gives the label distribution for each dataset.

5.6.2 How to choose the x_β and l_β parameters for F_β?

Using F_β requires to set the two parameters x_β and l_β. To choose these two parameters, one needs to answer to the following questions: At what *support size* a label subspace L is considered as over-represented? What is the speed of transfer from the normal representation state to the over-represented state? They are set thanks to the characteristics of the dataset. An automatic approach is to set x_β to the 85^{th} percentile value and l_β to the difference between the 85^{th} and 80^{th} percentiles. However, we advise to display the distribution of the frequency of the class labels (as shown in Figure 27) to correctly set up these parameters. The user must keep in mind that when $|supp(L)| = x_\beta + 2l_\beta$, $\beta \approx 0.02$ (L is considered thus over-represented and precision is fostered). Analogically, when $|supp(L)| = x_\beta - 2l_\beta$, $\beta \approx 0.98$ (L is considered normal and the F_β



(a) Reduced Olfaction (b) Yeast

Figure 28: Evolution of the quality of the result set of MCTS varying the number of iterations.

behaves almost as the $F1$ score).

5.6.3 Is MCTS able to consider WRF_β?

Since the search space related to our datasets are too large to employ an exhaustive search, we design two heuristic methods to experiment with DSSD and we study the diversity on target subspaces issue. We show that MCTS using RF_β allows to extract a diverse set of non redundant patterns if given enough computational budget (e.g., a given number of iterations). Figure 28 displays the evolution of the quality of the result set with MCTS when varying the number of iterations. We can note that, the more the iterations, the better the quality in the result set. Moreover, we use an exhaustive search in Olfaction to show that MCTS quickly converges to the quality of the result set obtained with the exhaustive search. However, since the search space of this dataset is too large, we randomly picked 2 attributes among the 82 attributes to make the exhaustive search tractable. Figure 28(a) shows that, given enough computational budget, MCTS converges to the quality of the result set obtained by the exhaustive search. From that, it is reasonable to employ MCTS with this new quality measure RF_β. Furthermore, on Figures 28(b), we can notice that the quality of the result set still increases with the number of iterations.

5.6.4 Which measure ensures the most diverse result?

We compare the diversity on the target subspaces in the result set. For that, we use MCTS with 100k iterations. We output the top-1000 subgroups when using the different quality measures $RAcc$, RF_1, RF_β, $WRAcc$, WRF_1 and WRF_β. Figures 29-30 display the results. We observe that, in general, $RAcc$ and $WRAcc$ covers few label subspaces compared to RF_1, WRF_1, RF_β and WRF_β. Indeed, in Olfaction, $RAcc$ covers twice less label subspaces than the others (see Figure 29(d)-(e)-(f)). However, on few datasets, the diversity on the target subspaces is almost the same, e.g., see Figure 29(d)-(e)-(f) for Genbase.

Besides, contrary to RF_1, the measures $RAcc$ and RF_β are able to homogeneously evaluate the subsets that are over-represented and non-over-represented. Thanks to the adaptive F_β, RF_β can either support the precision for the over-represented labels, or both precision and recall for non-over-represented labels. For the non-over-represented labels $RAcc$ does not also foster on the recall of the subgroups. However, with RF_1, the over-represented subgroups are rarely output in the result set because their recall is low when the precision is high and vice versa.

Moreover, the experiments with the weighted version of these measures lead almost to the

same result: WRF_β ensures a great diversity on the target subspaces and is able to characterize both over-represented labels and non-over-represented labels. However, the weighted factor fosters more on over-represented labels since the relative support size is greater. Note that when the distribution of the labels is not imbalanced, such as in Emotions, the measures behave in the same way, which is an expected behavior.

Finally, Figure 31 displays both the evolution of the diversity on the target space and the value of the quality measure of the subgroups in the result set varying the number k of output patterns. Figures 31(a)-(b)-(c) show that with *RAcc* or $WRAcc$ there is a few diversity in the target subspace whereas with RF_1, WRF_1, RF_β and WRF_β the diversity is high. Moreover, Figures 31(d)-(e)-(f) display that with the weighted version of these measure, the quality in the result set decreases faster than with the relative version. Indeed, the weighted factor makes the quality measure foster on over-represented subgroups and thus avoid the extraction of non-over-represented ones even if the precision and the recall is high.

5.6.5 Does RF_β also ensure the best diversity?

The previous subsection exhibits that RF_1, WRF_1, RF_β and WRF_β increase the diversity on the target space in the result set. In some cases, we showed that this diversity is also high with *RAcc* and $WRAcc$. However, in the previous chapter, we ensured that MCTS leads to a higher completeness of the result set. Indeed, we now consider some results obtained with a beam search. Figure 32 shows that for all the measures, except RF_β, there is a low diversity on the target space. Besides, except with RF_β, the beam search can not extract more than 300 diverse and non redundant subgroups. Clearly, RF_β provides the largest diversity in the result set.

5.6.6 Is the RF_β ranking the subgroups differently?

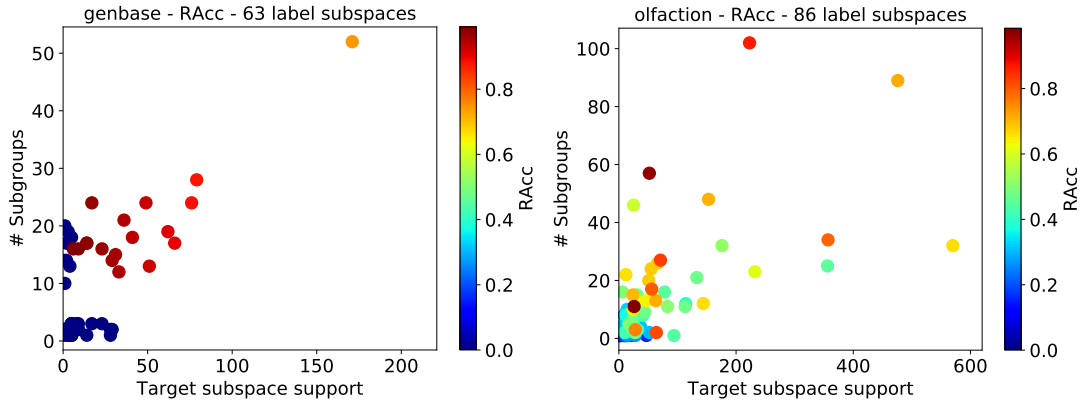
We experiment with the 6 different quality measures, to test if they are equivalent, i.e., they provide the same ranking of subgroups. Since it is not possible to use an exhaustive search, we cannot directly compare the ranking of the subgroups (most subgroups in a result set are not included in another result set). For that, we define an optimistic similarity measure between two result sets based on Jaccard coefficients: We compute the mean of the maximum of the Jaccard coefficients between a couple of subgroups in the two different result sets. Formally we compute $meanSim(S_1, S_2) = \frac{1}{|S_1|} \times \sum_{s_1 \in S_1} \max_{s_2 \in S_2} J(supp(s_1), supp(s_2))$ and then the similarity between two result sets S_1 and S_2 is:

$$maxSim(S_1, S_2) = \max\left(meanSim(S_1, S_2), meanSim(S_2, S_1)\right)$$

Figure 33 displays the matrix of $maxSim$ between the result sets obtained with the different quality measures in CAL500. Clearly, it shows that they do not share the same subgroups in their result set. Thus, the measures are not equivalent. The results are identical for the other datasets.

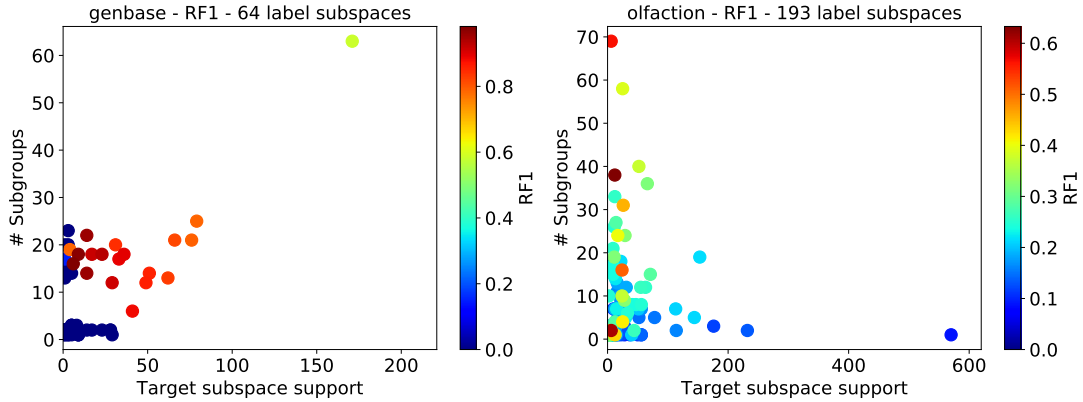
5.6.7 Does F_β dynamically adapt to the label frequency?

We said that for the over-represented labels, we want to support the precision, and that, for other frequencies of labels, we want to take into account both the precision and the recall of the subgroup. To assess the behavior of the measures, we experiment with the measures we use as models, namely *Acc*, F_1 and F_β. From that, we display the precision and the recall of the subgroups. Figure 34 shows two different views of the precision and the recall of the subgroup



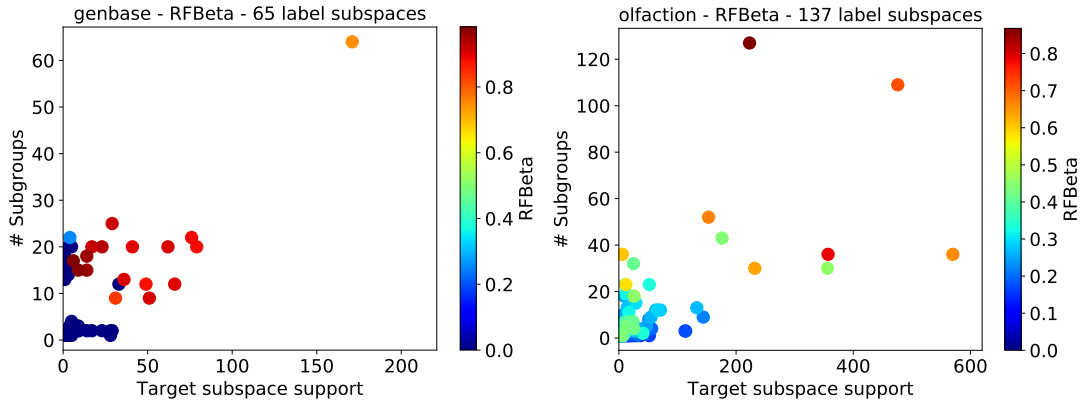
(a) Genbase: $RAcc$

(d) Olfaction: $RAcc$



(b) Genbase: RF_1

(e) Olfaction: RF_1

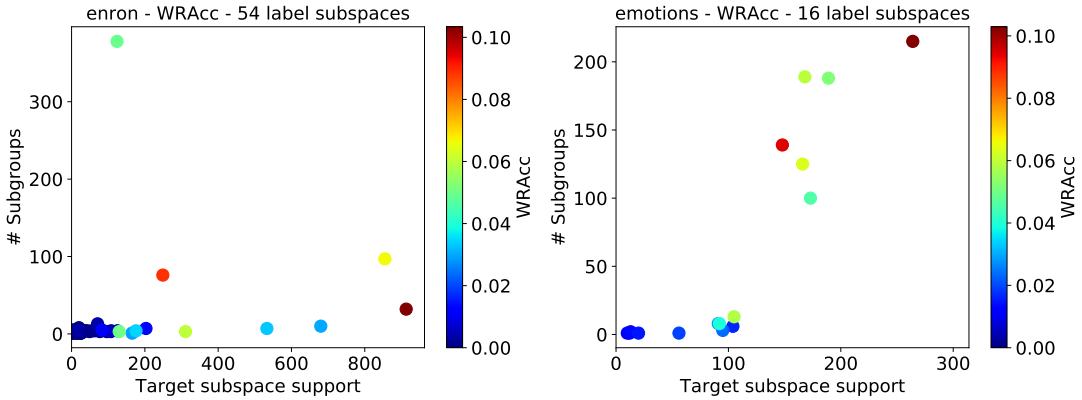


(c) Genbase: RF_β

(f) Olfaction: RF_β

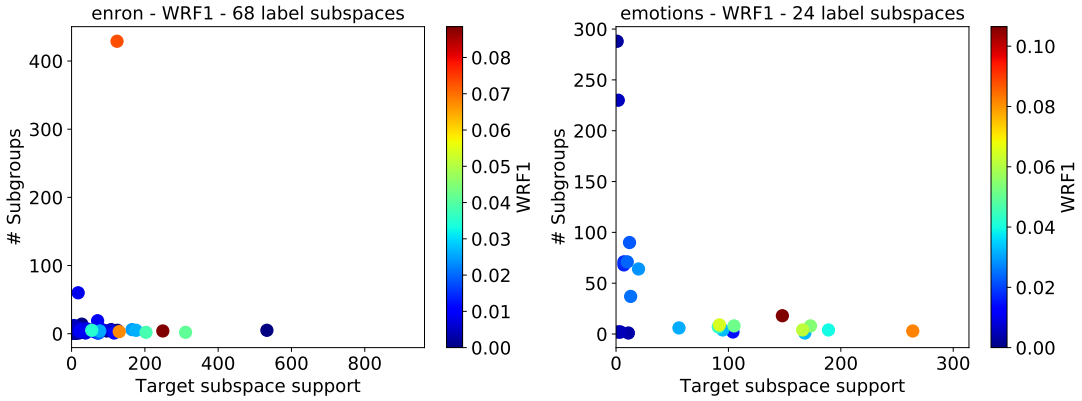
Figure 29: The quality measure and the support of the target subspace of the subgroups within result set obtained with MCTS using the Relative measures. The color of the points is the value of the quality measure of the subgroup given by the heatmap.

depending on the frequency of the subset of labels L they are related within the olfactory dataset. Figures 34(a)-(b)-(c) show the precision (triangles) and the recall (crosses) of the subgroup in the



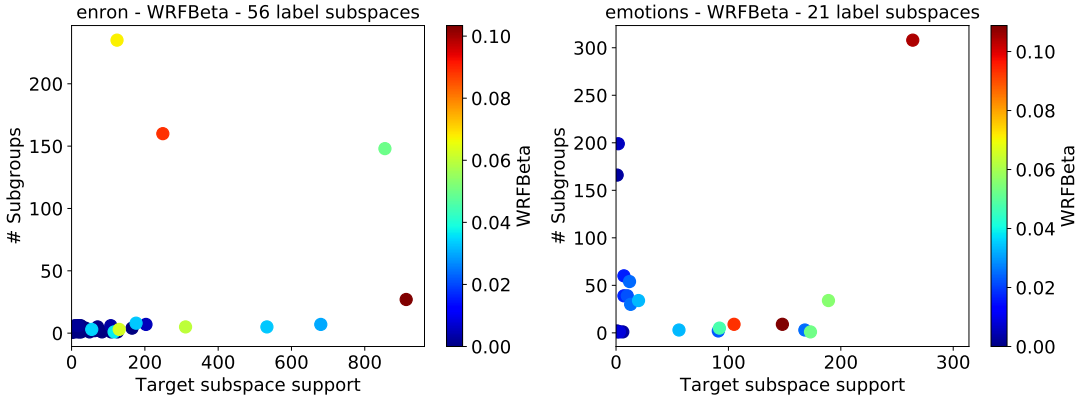
(a) Enron: $WRAcc$

(d) Emotions: $WRAcc$



(b) Enron: WRF_1

(e) Emotions: WRF_1

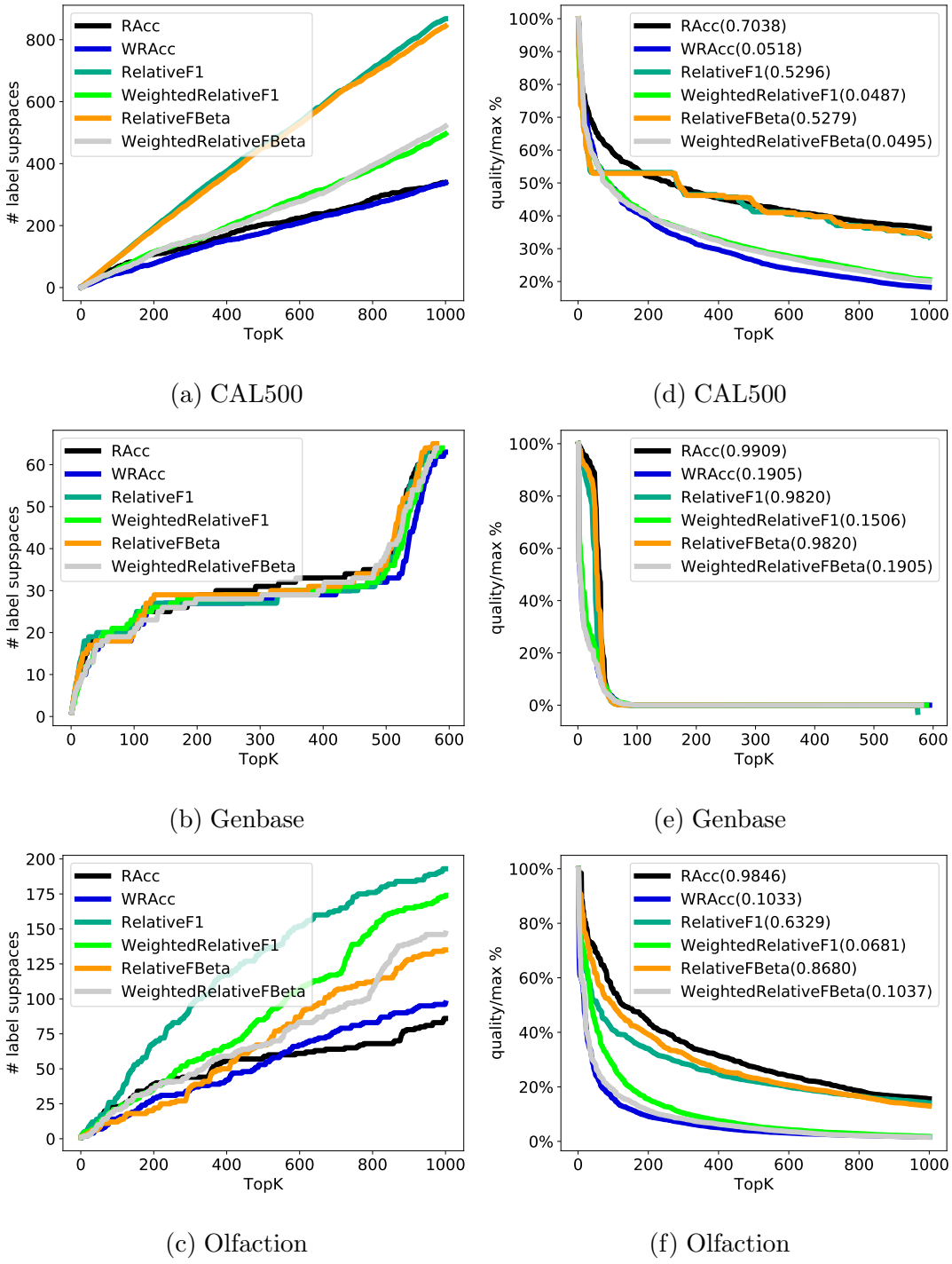


(c) Enron: WRF_β

(f) Emotions: WRF_β

Figure 30: The quality measure and the support of the target subspace of the subgroups within the result set obtained with MCTS using the Weighted Relatives measures. The color of the points is the value of the quality measure of the subgroup given by the heatmap.

result set. They also display the value of $\beta(supp(L))$. Clearly, for F_β, if the frequency of L is too high in the dataset, only the precision of the subgroup is fostered and the recall is low. But, if the frequency of L is *not* too high, both recall and precision are fostered. However, with Acc, only



(a) CAL500 (d) CAL500

(b) Genbase (e) Genbase

(c) Olfaction (f) Olfaction

Figure 31: Comparison between the measures on the extracted top-K. Evolution of the quality measure in the top-K.

the precision is fostered, the recall is always low whatever the frequency of L. Concerning F_1, we support both the precision and the recall, but there are less subgroups in the result set that are related to labels for which the frequency is high because the F_1 score is low.

Figures 34(d)-(e)-(f) present another point of view to illustrate this behavior. The subgroups

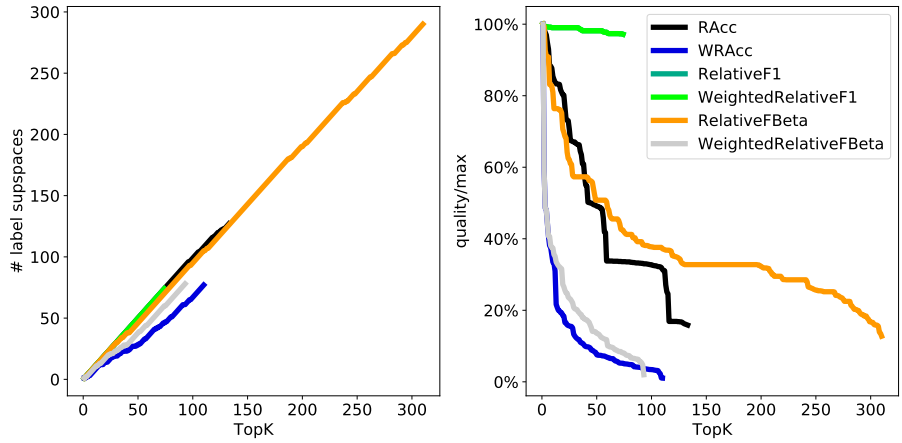


Figure 32: Comparison with a beam search on the Olfaction dataset. .

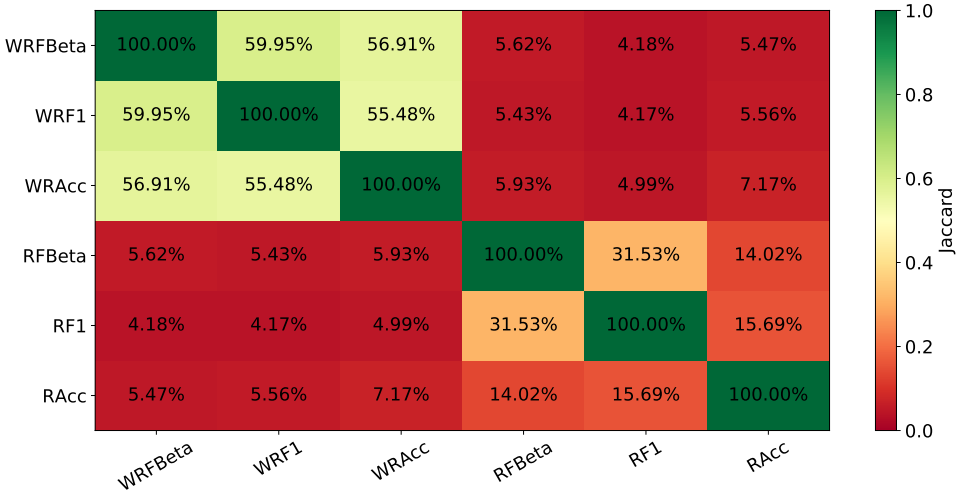
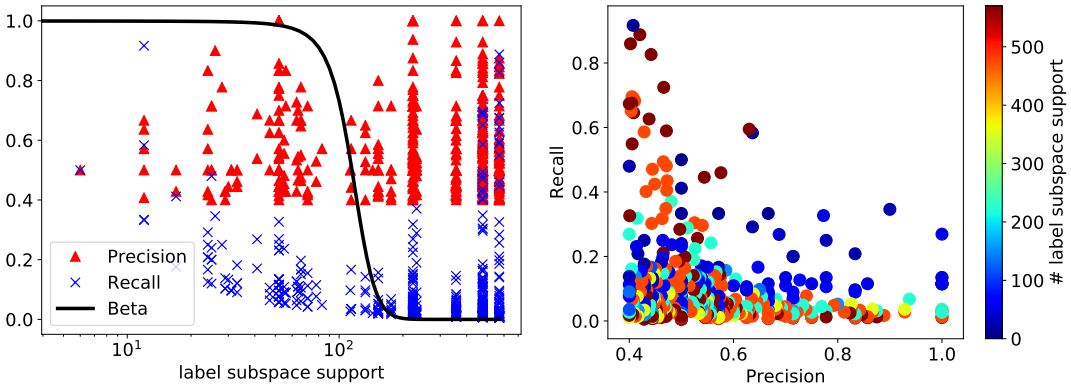


Figure 33: The matrix of the similarity of the result set using different quality measures.

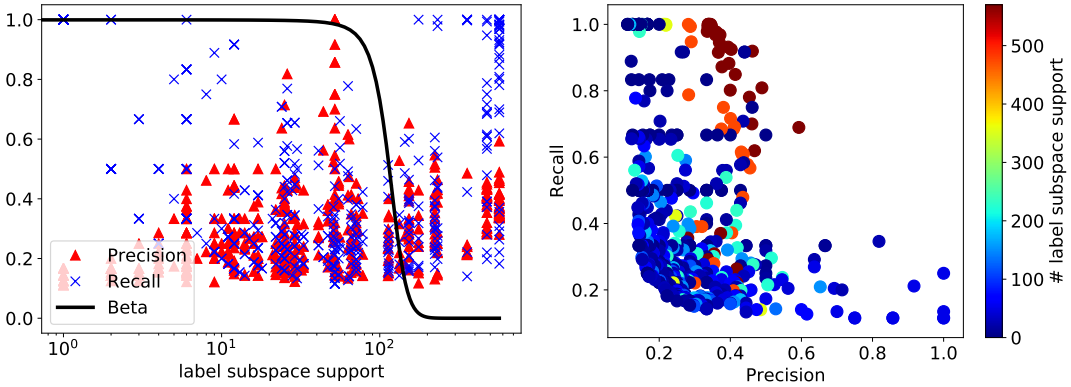
in the result set are plotted in the precision/recall space. Moreover a heatmap is used to show the frequency of the subset of labels L to which the subgroups are related: The points in red are subgroups related to over-represented labels, and the points in blue are subgroups related to under-represented labels. Clearly, Acc does not foster on the recall. Besides, F_1 extracts few subgroups related to over-represented labels (many blue points). Finally, F_β behaves as expected: (i) Both over-represented and under-represented labels are covered, i.e., the diversity in the target space is high ; and (ii)for over-represented labels the precision is fostered, and for other labels both recall and precision are taken into account.

5.6.8 Synthesis of the experiments

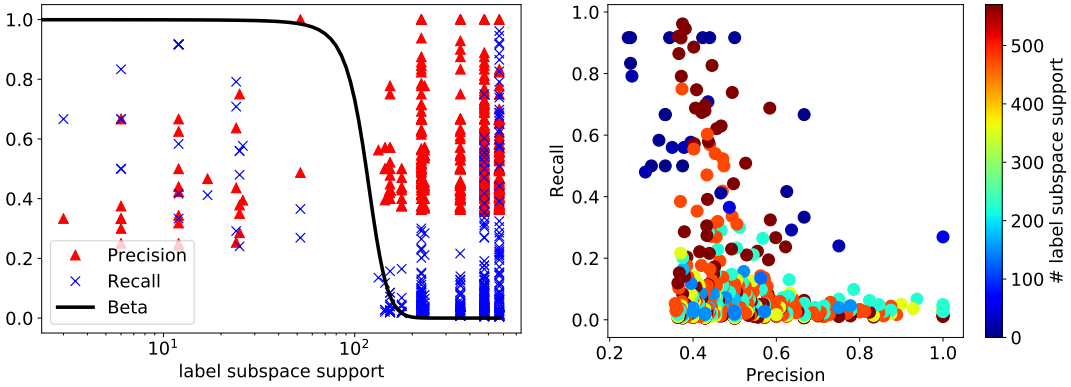
The parameters x_β and l_β for the F_β measure can be easily set thanks to the distribution of the label frequency. In general, our results suggest that exhaustive search is not tractable because the search space of the datasets is too large, and thus heuristic explorations are needed. MCTS is able to quickly find interesting subgroups with RF_β and WRF_β. Moreover, the results support the idea that RF_β provides a better diversity on the target subspace using either MCTS or a beam



(a) *Acc* (d) *Acc*



(b) F_1 (e) F_1



(c) F_β (f) F_β

Figure 34: Precision and recall of the models of the extracted subgroups on Olfaction.

search. All these quality measures focus on different subgroups and thus there is no equivalence between each other. Finally, as expected, the F_β measure enables to foster the precision for over-represented labels and to support both recall and precision for other label subsets.

5.7 Conclusion

Motivated by a problem in neuroscience and olfaction, we proposed an original subgroup discovery approach to mine descriptive rules characterizing specifically subsets of class labels. For that matter, we revisited the *diverse subgroup set discovery* problem within the *exceptional model mining* framework: Each subgroup can be evaluated in different *target subspaces*, and a quality measure helps to take into account the distribution of the labels overs the dataset. This measure is also effective in heuristic search, e.g., with an enumeration based on Monte Carlo Tree Search, as the results are more diverse, both on the subgroup description and the target subspaces. We showed the effectiveness of this method through a large set of experiments.

Chapter 6

Application in neuroscience for the study of the olfactory percept

6.1 The Structure-Odor Relationship

Around the turn of the century, the idea that modern, civilized human beings might do without being affected by odorant chemicals became outdated: The hidden, inarticulate sense associated with their perception, hitherto considered superfluous to cognition, became a focus of study in its own right and thus the subject of new knowledge. It was acknowledged as an object of science by Nobel prizes (e.g., [37] awarded 2004 Nobel prize in Physiology or Medicine); but also society as a whole was becoming more hedonistic, and hence more attentive to the emotional effects of odors. Odors are present in our food, which is a source of both pleasure and social bonding; they also influence our relations with others in general and with our children in particular. The olfactory percept encoded in odorant chemicals contribute to our emotional balance and well-being: Olfactory impairment jeopardizes this equilibrium.

Neuroscience studies revealed that odor perception results from a complex phenomenon rooted in the chemical properties of a volatile molecule (described by multiple physicochemical descriptors) further detected by our olfactory receptors in the nasal cavity. A neural signal is then transmitted to central olfactory brain structures [113]. At this stage, a complete neural representation, called "odor" is generated and can then be described semantically by various types of perceptual quality (e.g., fruity, floral or woody). While it is generally agreed that the physicochemical characteristics of odorants affect the olfactory percept, no simple and/or universal rule governing this Structure Odor Relationship (SOR) has yet been identified. Why does this odorant smell of roses and that one of lemon? Considering that the totality of the odorant message was encoded within the chemical structure, chemists have tried to identify relationships between chemical properties and odors. Topological descriptors, eventually associated with electronic properties or molecular flexibility were tentatively connected to odorants descriptors. However, it is now quite well acknowledged that structure-odor relationships are not bijective. For example, very different chemicals trigger a typical "camphor" smell, while a single molecule, the so-called "cat-ketone" odorant, elicit two totally different smells as a function of its concentration [50]. At best, such SOR rules are obtained for a very tiny fraction of the chemical space, emphasizing that they must be decomposed into sub-rules associated with given molecular topologies [55]. As such, this lack of bijective relationship must be handled. It suggests that a simple, universal and perfect rule does probably not exist, but instead, a combination of several sub-rules should be put forward to encompass the complexity of SOR. In this chapter, we describe our data science approach to

advance the state of the art in understanding the mechanisms of olfaction and to support the emergence of new hypothesis.

Therefore, we have been combining neuroscience, chemistry and data mining. Indeed, data-mining methods can be used to answer the SOR discovery problem, either through the building of predictive models or through rules discovery. One obstacle to this is that olfactory datasets are complex (i.e., several thousand of dimensions, heterogeneous descriptors, multi-label, imbalanced classes, and non robust labelling) and, we suffer from a lack of data-centric methods in neuroscience that would be suitable for this level of complexity. The main aim of our study has been to examine this issue by linking the multiple molecular characteristics of odorant molecule to olfactory qualities (fruity, floral, woody, etc.) using a descriptive approach (pattern mining). Indeed, a crowd-sourced challenge was recently proposed by *IBM Research* and *Sage* called *DREAM Olfaction Prediction Challenge* [65]. The challenge resulted in several models that were able to predict especially pleasantness and intensity and 8 out of 19 olfactory qualities ("garlic", "fish", "sweet", "fruit", "burnt", "spices", "flower" and "sour") with an average correlation of predictions across all models above 0.5 [91]. These findings are timely and interesting because they show the existence of a predictive relationship between certain olfactory qualities and the physico-chemical properties of odorant molecules. Nevertheless, to go further into the understanding of the stimulus-percept issue in olfaction, it is important to explain and thus to isolate physico-chemical rules allowing describing these olfactory qualities. Identifying such physico-chemical rules characterizing odor quality opens a field that is still little explored in neuroscience research, shedding new light on olfactory system functioning.

Neuroscientists are faced to a series of issues including (i) the lack of large databases containing sufficient number of stimuli described by both physicochemical properties and olfactory qualities, (ii) the large number of dimensions – several thousands – needed to describe the molecules and, above all, (iii) a lack of data-processing methods suitable for this level of complexity. The present chapter proposes to solve these issues by - first - setting up a large database containing more than 1500 odorant molecules described by both of physico-chemical properties and olfactory qualities. Secondly, we make use of such complex heterogeneous data with the aim of deriving new neuro-scientific knowledge or hypotheses such as descriptive models. This is where our new EMM instance based on the MCTS exploration fully makes sense as molecules are associated to several odors, with a highly skewed distribution and a very large search space.

6.2 An original olfaction dataset

One prominent methodological obstacle in the field of neuroscience concerns the absence of any large available database (>1000 molecules) combining odorant molecules described by two types of descriptors: Perceptual ones such as olfactory qualities (scent experts defining a perceptual space of odors), and chemical attributes (chemical space). The dataset provided by the IBM challenge [91] is a clinical one, i.e., odorant molecules were not labeled by scent experts. To tackle this issue, the neuroscientists have recently created a new dataset based on the olfactory percept of scent experts. Basically, they gathered the data from two different sources: (i) The *Dragon 6* software (available on `talete.mi.it`), and (ii) the well known atlas Arctander [9]. The first one provides the values of almost $5,000$ physico-chemicals properties (e.g., the *molecular weight* or the *number of carbon atoms*) for thousands of chemical molecules. The latter is an atlas created by scent experts that maps $1,689$ molecules to subset of odors among 74 olfactory qualities such as *Fruity*, *Vanilin*, or *Camphor*. From such raw data, the neuroscientists pre-processed the data to keep only the molecules that are present in both datasets. Moreover, the chemists identified

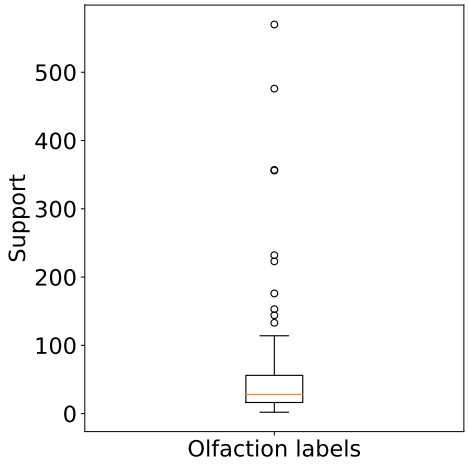


Figure 25: Olfaction data label distribution.

and pointed out a set of 82 physico-chemical properties that have been known or guessed to be discriminant in SOR. Thus, the final data is made of 1,689 molecules described by 82 physico-chemical properties and associated to at average 2.88 odors among the 74 olfactory qualities. As such, and to the best of our knowledge, the present database is one of the very few in the field that enable quantification and qualification of more than 1,500 molecules at both, perceptual (neurosciences) and physicochemical (chemistry) levels. The distribution of the 74 olfactory qualities is illustrated in Figure 25.

6.3 h(odor): An interactive tool to elicit hypotheses on SOR

We developed an interactive Web application that enables the experts of the domain (e.g., neuroscientists and chemists) to directly interact with the algorithm to guide the exploration of the search space. In other terms, they can use their knowledge to exploit the subgroups that seem to be interesting or to explore some parts of the search space that have been ignored so far. To interact with the algorithm, the user can like or dislike some temporary rules. This feed-back is used to propose new rules that can be more interesting for a given user. The application, called *h(odor)*, is available online with a video tutorial supporting the use case at `http://liris.cnrs.fr/dm2l/hodor/`.

6.3.1 System architecture

A core module (server) is contacted by a client (Web interface) to initiate the mining algorithm with the given parameters. This core module allows the user to interact/guide the exploration based on the likes/dislikes of the user.

The Core Module

This is the back-end of the *h(odor)* application. Based on *NodeJS*, the *Core Module* is the gateway between the user and the ELMMut algorithm (see next subsection): It is in charge of the interaction. For that, JSON data are sent to and received from the *SD algorithm* through sockets

thanks to a dedicated communication process. Moreover, this module controls the UI to display results extracted from the *SD Algorithm* and collects the user preferences (like/dislike).

The User Interface (UI)

The front-end of the application, based on *Bootstrap* and *AngularJS*, enables the user to select the parameters of the *SD Algorithm* and to run it. Once the subgroups of the first level of the *beam search* are extracted (the algorithm is paused waiting the user preferences), the UI displays these subgroups and the user can like/dislike some of them: The liked subgroups are exploited in the next iteration of the algorithm, whereas disliked subgroups are removed from the explored search space. When the algorithm finishes, the UI displays the results.

6.3.2 Use case: Eliciting hypotheses on the Musk odor

We develop a use case of the application from an end user perspective, typically a neuroscientist or a odor-chemist that seeks to extract descriptive rules to study the Structure-Odor Relationships. In this scenario, we consider that the expert wishes to discover rules involving at least the *musk* odor.

1- Algorithms, parameters and dataset selection

In the *Algorithms* section of the left hand side menu, the user can choose the exploration method and its parameters. In this use case, we exploit the ELMMut algorithm. It implements a beam search strategy to extract subgroups based on a quality measure. We plan to add the mcts4dm algorithm: It requires to implement the interactive process within the MCTS exploration. Once the exploration method is chosen, we have to select the olfactory dataset as introduced in the previous section, and choose to focus on the *musk* odor. We decided to set the size of the beam to 50 (the exploration is quite large enough) and the minimum support threshold to 15 (since $supp(Musk) = 52$, at least the subgroups have to cover 30% of the musk odorants). Other parameters are fixed to their default values.

2- Interactive running steps

When the datasets and the parameters have been fixed, the user can launch the mining task clicking on the *Start mining* button. When the first step of the *beam search* is finished, the *SD Algorithm* is paused and the subgroups obtained at this step are displayed to the user. The interaction view in the front-end presents the olfactory qualities involved at this level of the exploration (see Figure 35). Each subgroup is displayed in a white box with the current descriptive rule on the physicochemical descriptors and some quantitative measures. For each subgroup box, the user can select in the top right corner if she likes/dislikes this subgroup. For example, at the first step, the application displays the subgroups extracted at the first level for the *Musk* odor. As it is a known fact in chemistry that the *musk* odor involves large molecules, we *like* the subgroup which description is $d = [238.46 \leqslant MW \leqslant 270.41]$. After that, we keep on exploring by clicking the *Next* button. Another interactive step begins, but the expert has no particular opinion so she can jump to the *next* level.

3- Analysis of the results

Once the algorithm finishes (the quality measures cannot be improved), we can study the table of results. For example, the description of one of the best extracted subgroups s is: $[238.46 \leqslant MW \leqslant$

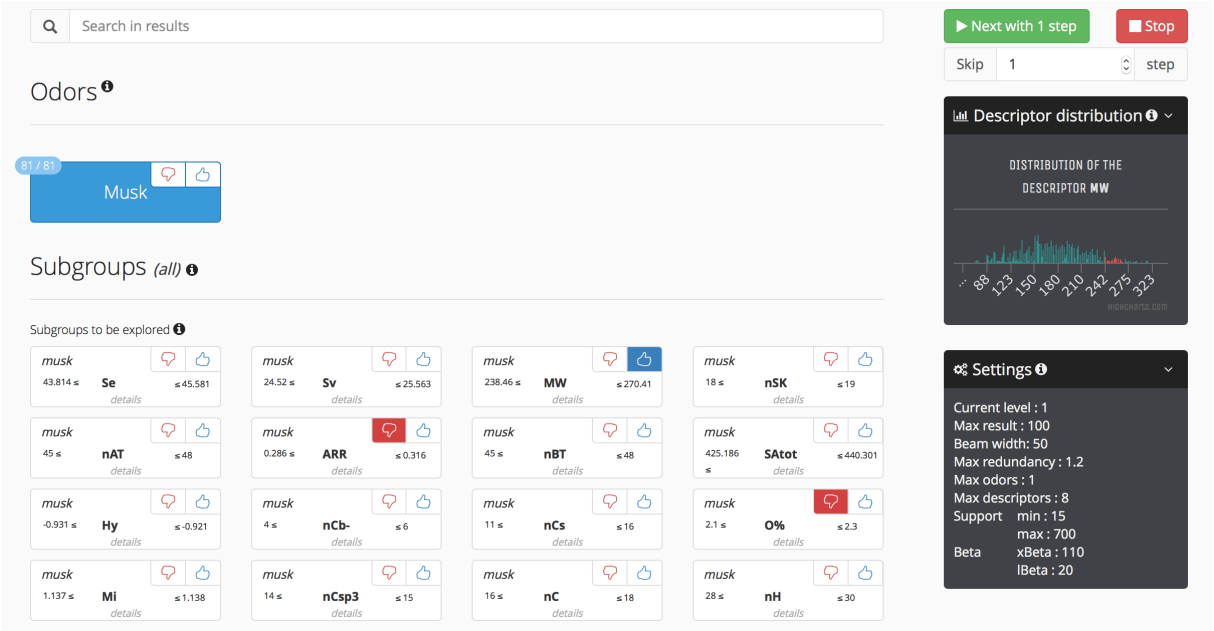


Figure 35: The interaction view of the application. For each step of the beam search, the algorithm waits for the user's preferences (like/dislike). The subgroups are displayed into white boxes. On the right part, complementary information is displayed: Part of value domain of a chosen restriction on a descriptor, and parameters of the run.

$270.41][-0.931 \leqslant Hy \leqslant -0.621][2.714 \leqslant MLOGP \leqslant 4.817][384.96 \leqslant SAtot \leqslant 447.506][0 \leqslant nR07 \leqslant 0][0 \leqslant ARR \leqslant 0.316][1 \leqslant nCsp2 \leqslant 7]$ that involves large odorants. In this subgroup, the interval restriction on the molecular weight MW corresponds to high values, and thus to large molecules. The goal of the *h(odor)* application is to confirm knowledge and to elicit new hypotheses for the SOR problem. In the case of s, the neuroscientists are interested in understanding why these descriptors (excepted the molecular weight) are involved in the $Musk$ odor. With the *h(odor)* tool, we can also proceed to rule combination (see Figure 36). Indeed, our experts suggest that a simple, universal and perfect rule does probably not exist, but instead, a combination of several sub-rules should be put forward to encompass the complexity of SOR. The user can manually select several rules in the result set to evaluate the quality of their combination on a graph. Also, *h(odor)* enables to automatically computes the best combinations of rules w.r.t. the precision and the recall of the combination over the odors[10].

6.4 Eliciting new hypothesis on the SOR problem

We presented the interactive Web application *h(odor)* that enables the user (e.g., the neuroscientist or the chemist) to guide the exploration of the search space. Our experts keep on experimenting with this application. However, *h(odor)* does not embed the MCTS4DM algorithm yet, though it would provide better results than beam search based approaches. Thus, the experts are also analyzing and interpreting the rules provided by MCTS4DM to study these raw results before to

[10]Besides, note that the *h(odor)* application enables to save all the choices taken by the different users. Indeed, the application stores all the user's actions into log files. The goal here is to use these log files to learn user preferences, not only for a single run of the algorithm [64] but for all experiments performed by the users. This kind data (choices made by experts) is hard to collect by simply asking experts and will be explored in future work.

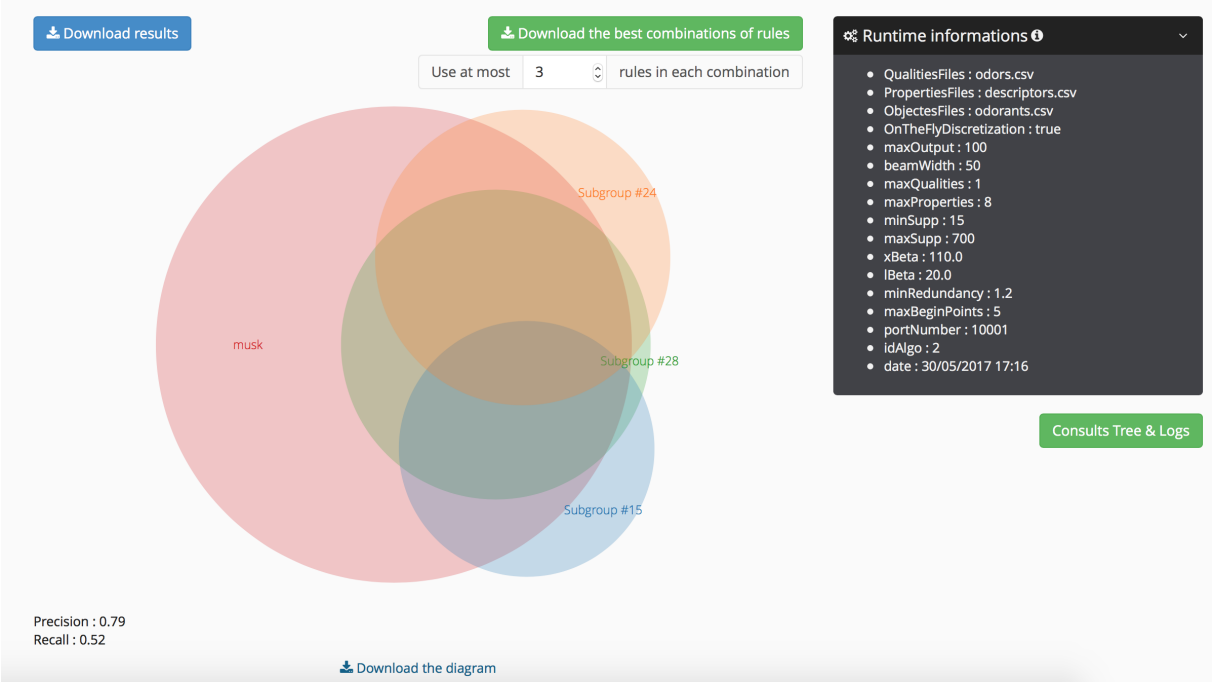


Figure 36: The rule combination that covers the musk odor.

interact with the algorithm. In this section, we provide some feedback and interpretations given by the experts on the results computed by MCTS4DM.

6.4.1 Identification of relevant physicochemical attributes

We consider the experiment on the olfactory dataset (introduced in this chapter) when we use the RF_β score, $minSupp = 30$. A relevant information for neuroscientists and chemists concerns the physicochemical attributes that appear in the descriptive rules. As showed in [92], the sum of atomic van der Waals volumes, denoted as Sv, is discriminant with regard to the hedonism of an odor, and the higher Sv, the more pleasant an odor. Moreover, the higher the rate of nitrogen atoms ($N\%$), the less pleasant an odor is, consistent with the idea that amine groups ($-NH_2$) are associated with bad odors (such as cadaverine or putrescine). Based on these observations, we find subgroups related to either the *Floral* or *Fruity* quality that are characterized by a special range of values with regard to Sv and $N\%$. For example, $s_1 = \langle [3.0 \leqslant N\% \leqslant 33.3]\ [2.7 \leqslant O\% \leqslant 33.3]\ [0.0 \leqslant nR05 \leqslant 1.0]\ [nRCN = 0.0], \{Fruity, Grape\}\rangle$ and $s_2 = \langle [6.57 \leqslant Sv \leqslant 43.17]\ [1.0 \leqslant nN \leqslant 3.0]\ [1.0 \leqslant nHDon \leqslant 2.0], \{Floral\}\rangle$ are output subgroups. The quality measure of s_1 is 0.23 with a precision of 0.16 and a low recall of 0.5. For s_2, its quality measure is up to 0.37, its precision is 0.65 and its recall is 0.06. The first subgroup contains in its description the $N\%$ attribute associated to a very low percentage, and s_2 includes the Sv attributes with a range of values that corresponds to its higher values. In general, the quality $Musk$ is associated with large and heavy molecules: The molecular weight (MW) of these molecules is thus high. In the output subgroups, most of those associated to the musk quality include in their description the MW attribute with high values, or any other attribute that is positively correlated with MW, such as $SAtot$. For example, $s_3 = \langle [27.03 \leqslant MW \leqslant 297.3]\ [1.0 \leqslant nBM \leqslant 18.0]\ [nR03 = 0.0]\ [0.0 \leqslant nCp \leqslant 7.0]\ [5.0 \leqslant nCrs \leqslant 16.0]\ [nHDon = 0.0], \{Musk\}\rangle$ with a quality measure of 0.2 (precision: 0.27, recall: 0.25) is about molecules with it molecular weight between 27.03 and 297.3. Moreover, when the quality $Musk$ is combined

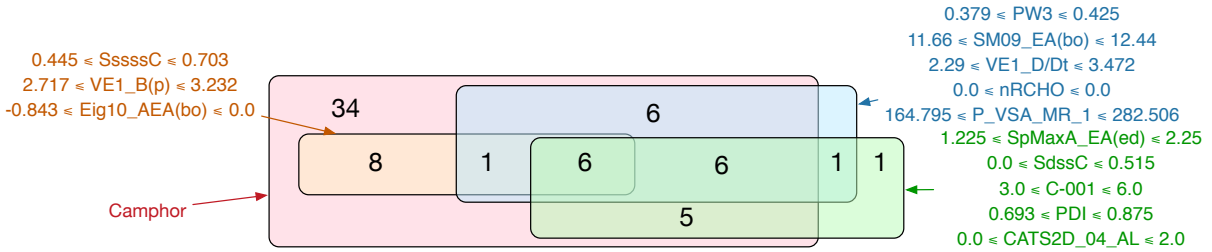


Figure 37: The support of three groups involving the *camphor* odor.

with the quality *Animal*, we still have a high molecular weight but there are other attributes with specific range of values: $s_4 = \langle [168.808 \leqslant SAtot \leqslant 464.918][6.0 \leqslant nCar \leqslant 16.0][3.261 \leqslant MLOGP \leqslant 4.593][0.0 \leqslant nCsp2 \leqslant 7.0], \{Musk, Animal\}\rangle$. This latter topological attribute is consistent with the presence of double bonds (or so-called $sp2$ carbon atoms) within most of the musky chemical structures, to provide some hydrophilicity.

6.4.2 Providing relevant knowledge on the process of the olfactory percept

Another important information brought by such findings lies in the fact the SOR problem should be viewed and studied thanks to a "multiple description" approach rather than a "one rule for one quality" approach (i.e., bijection). Indeed, a number of odor qualities were described by very specific rules. For example, 44% of the molecules described as *camphor* can be described by 3 rules physicochemical rules, with a very low rate of false positives (0.06%; molecules being described by the physicochemical rule, but not described perceptively as *camphor*). Similar patterns were observed for other qualities: e.g., *mint* (3 descriptive rules; 32% of the molecules described as *mint*; 0.06% of false positives), *ethereal* (3; 35%; 0%), *gassy* (3; 36%; 0.36%), *citrus* (3; 42%; 0.24%), *waxy* (3; 43%; 0%), *pineapple* (3; 48%; 0%), *medicinal* (3; 49%; 0.30%), *honey* (4; 54%; 0.06%), *sour* (3; 56%; 0.36%). Focusing on these qualities, this confirms that a universal rule cannot be defined for a given odorant property, in line with the extreme subtlety of our perception of smells. For example, looking in more details on the produced rules for Camphor (see Figure 37), it appears that one rule is mostly using topological descriptors, while the second rather uses chemical descriptors. The third rule has a combination of these two.

6.5 Perspectives in neurosciences and chemistry

The present findings provide two important contributions to the field of neurosciences and chemosensation. First, although the SOR understanding seems to be illusory for some odor qualities, our approach suggests that there exist descriptive rules for some qualities that also highlight the relevance of some physicochemical descriptors (Sv, MW, etc.). Second, the present model confirms the lack of bijective (one-to-one) relationship between the odorant and the odor spaces and it emphasizes that several sub-rules should be taken into account when eliciting structure-odor relationships. From these findings, experts in neurosciences and chemistry may generate the following new and innovative hypotheses in the field: (i) explaining inter-individual variability in terms of both behavioral and cognitive aspects of odor perception, (ii) explaining stability in odor-evoked neural responses and (iii) correlating the multiple molecular properties of odors to their perceptual qualities. Our collaborators are preparing a paper for the journal in neuroscience *PLOS One - Computational Biology*.

Chapter 7

Conclusion and perspectives

7.1 Summary

Motivated by a problem in neuroscience and olfaction, we designed a data science approach based on the knowledge discovery in databases process (KDD) to elicit new hypotheses on the structure-odor relationship. For that purpose, we studied subgroup discovery (SD) and its generalization exceptional model mining (EMM) to support the discovery of supervised rules that distinguish target labels.

In presence of large data, such as the olfactory dataset that contains lots of numerical descriptors, an exhaustive exploration of the search space is unfeasible. Thus, some heuristic approaches have been used, e.g., beam search, evolutionary algorithms and sampling methods. However, they are not able to propose a high completeness in the result set, i.e., containing many different enough local optima. We advocate for the use of MCTS for pattern mining: An aheuristic exploration strategy leading to *"any-time"* pattern mining that can be adapted with different measures and policies. The experiments show that MCTS and our MCTS4DM algorithm provides a much better completeness in the result set than existing heuristic approaches. Interesting subgroups are found in a reasonable amount of iterations and the quality of the result iteratively improves. In addition, we show that numerical attributes are handled without using greedy discretization either within a pre-processing task or by means of an on-the-fly setting during the enumeration. We enumerate all the possible intervals to ensure that MCTS4DM converges to an exhaustive search if given enough time and memory.

Moreover, we proposed an original subgroup discovery instance to mine descriptive rules characterizing specifically subsets of class labels. For that matter, we revisited the *diverse subgroup set discovery* problem within the *exceptional model mining* framework: Each subgroup can be evaluated in different *target subspaces*. A subgroup does not derive one model but several models: One model is built on each subset of class labels. It induces a huge increase of the search space size that can still be handled by the MCTS enumeration. As we face a high variance in the frequency of the class labels in the olfactory dataset (some odors are much more frequent than others), we define a new model F_β. It employs a function β that dynamically either fosters on the precision of the subgroups w.r.t. the target subspace if this latter is over-represented is the data, or that converges to the F_1 value score for other target subspaces. We derive several quality measures to compare two instances of our model. This EMM instance is also tractable for a MCTS-based search, as the results are more diverse, both on the subgroup description and the target subspaces. We showed the effectiveness of this method through a large set of experiments.

Finally, the combination of both the enumeration based on MCTS and the new EMM instance

is used to elicit new hypotheses on the structure-odor relationship in our olfactory application in collaboration with neuroscientists and chemists. The results suggest new understandings about the links between the physico-chemical properties of a molecule and its odor. The experts of the domain are still working on the results provided by both our algorithm MCTS4DM and our interactive Web application *h(odor)*[11].

7.2 Perspectives

The exploration based on Monte Carlo tree search is essentially used in the artificial intelligence domain to solve games and planning problems. To the best of our knowledge, it is the first attempt to revisit it for pattern mining issues. Thus, it remains many opportunities to improve and to adapt it for other pattern mining tasks than subgroup discovery and exceptional model mining. In this section, we highlight the main issues we plan to study either for the short-term or a long-term perspective.

7.2.1 Improvements of our algorithm MCTS4DM

We have showed that our algorithm MCTS4DM is able to quickly find a diverse set of patterns with a high quality measure. However, we notice that the memory linearly increases with the number of objects in the data and the number of iterations. This is due to the enumeration of the several projected databases when specializing a description. Indeed, each node of the search tree stores a bitset for the extent of the subgroup related to this node: The more the iterations, the more the nodes in the tree, and thus the more the bitsets. To reduce the impact of both the number of objects and the storage of bitsets with the generation of a node, we are working on a new implementation method that only stores one bitset for the extent for all the nodes. For each iteration, it consists of re-processing the refinements of each selected node during the SELECT method from the root to the final selected node s_{sel}. This enables to reduce the memory usage but it can be a bit more time consuming since we have to compute the extent and the description of each selected node. However, the implementation of structures storing the data in an efficient way (e.g., ordering the values of numerical attributes and using pointers to the objects that take a specific value) enables to reduce the runtime. In addition to this algorithmic improvement, we plan to implement an interactive version of MCTS4DM to extend the interactive Web application *h(odor)*. We also think about using parallelisation for our algorithm to improve its efficiency. Besides, we may also handle the high branching factor we faced in several dataset. We discuss these three improvements in the following.

Interactive version

Our *h(odor)* application relies on an interactive algorithm allowing the experts to use their knowledge to guide the exploration of the search space. Beam search algorithms are easy to turn interactive by stopping after each level of the exploration to propose to the expert to select which of the subgroups would be extended at the next iteration. However, making MCTS4DM interactive would require to study when the algorithm has to be paused to take the expert's feedback, which temporary patterns to output to the user and then how to incorporate the feedback into the search tree. The first point can be solved by pausing the algorithm, say each $1,000$ iterations, or to allow the user to manually interrupt the algorithm when he/she wants. Solving the second

[11] Our collaborators are preparing a submission to PLOS One Computational Biology

issue appears more difficult. After N iterations, MCTS4DM has computed at least N patterns and eventually those that are stored during the memory policy. The question is thus, which patterns have to be output to the user to allow him/her to like/dislike. We can randomly sample k subgroups among all the computed ones, or we can sample them based on their quality measure. The third point concerns the use of the expert's feedback in the search tree. A naive solution would be to bias the aggregated value of rewards $Q(s)$ of a liked/disliked subgroup s to improve/decrease its UCB value. However, it also requires to bias all of its parents to reach it from the root if it is liked and conversely avoid it to be selected if it is disliked. In addition, the choice of the node to expand can also be biased to foster on user's preferences. Similarly, during the simulation, the refinements may no longer be randomly picked but biased by the expert's feedback.

Parallelisation

The characteristics of the MCTS algorithm, where each iteration is independent w.r.t. the others, make this approach tractable for parallelisation. Parallelisation can provide great advantages by performing several iterations over the tree in the same time, and thus be able to quickly and efficiently enumerate the search space. There exists three main approaches to employ parallelisation for MCTS algorithm (see Chapter 3): The leaf parallelisation, the root parallelisation and the tree parallelisation [43, 46]. In our settings, we think that the most usefull approach is leaf parallelisation since several parallel simulations are run from the expanded node s_{exp}. It enables to get a more precise overview of the search space rooted by s_{exp}. Besides, this approach does not require to use a mutex on the search tree and the pool of simulations are run together. However, its main drawback is that the iteration is over when *all* the simulations are completed: The runtime depends on the longer simulation. Another approach would be to split the search space into different parts. In each part a MCTS is run. This method would avoid to put a mutex on the tree but it will not respect the global exploration/exploitation trade-off but instead a local trade-off in each part of the search space: Some less interesting parts would be as exploited as interesting parts. Finally, we are interested to launch several MCTS instances in the search space rooted in different parts of the lattice. Basically, as a pre-processing task, we can project the pattern language into a less complex one (e.g., from sequences to itemsets) in order to find interesting patterns in this less expressive language that would be used as seeds to launch several MCTS instances within the same search tree.

Handling the high branching factor

A well known problem in the MCTS method is that it tends to explore much more than it exploits. Indeed, since MCTS expands all the children of a node before exploiting one of its children, it requires lots of iteration to explore the search space in depth. This is due to the high branching factor. For that the progressive widening method has been proposed [45, 49]. Basically, the number of children that have to be expanded before selecting a child depends on the number of visits $N(.)$. The keypoint in this method is the choice of the child to expand. Indeed, it is better to expand with the best children at the beginning since it requires more iterations to expand all the children. The choice of the child to expand can be done with the Rapid Action Value Estimator (RAVE) [73, 42]. Moreover, in addition to RAVE, we may take into account redundancy in this step: The choice of the child to expand can be based on both RAVE and a measure that states if the considered child is similar to an existing node in the search tree or if it is somehow different from the computed subgroups. Thus, the child having the best RAVE and that is not similar to other patterns has more chance to be chosen than other children.

7.2.2 Handling complex pattern mining issues

As future work, we are also interested in solving some pattern mining problems with MCTS. In this thesis, we studied supervised rules discovery with subgroup discovery and exceptional model mining. We considered boolean, nominal and numerical attributes. The subgroups are evaluated with one quality measure. However, in pattern mining, there exists different settings and constraints which we plan to solve with our framework based on MCTS. The first one consists in considering more expressiveness in the pattern language. The second is to take into account several quality measures to evaluate a pattern.

Toward more expressiveness in the pattern language

In pattern mining, many pattern languages have been defined and studied: From itemsets to graphs, sequences, or polygons. To handle different pattern languages, it is required to define formally both the search space and the refinement operators. For instance, for the case of sequential pattern mining, several algorithms have been proposed to explore the search space in different ways [79]. The traditional way to deal with sequential patterns is to use sequence extensions or itemset extensions with projected databases such as *PrefixSpan* algorithm [122]. This is the enumeration method we employed in our previous works on sequential pattern mining to search for imbalanced strategies in electronic sport data. However this enumeration principle is not suitable for a sampling method such as MCTS, since it would not sample the search space uniformly (as for the lectic order itemset mining we discussed in this thesis). The naive way consisting in extending a sequence with an item that can be appended in any existing itemsets or in new ones is possible but the branching factor would burst. Besides, MCTS has already been applied to enumerate convex polygons [16] and the results suggest that MCTS can handle efficiently more complex pattern languages by defining correct strategies. In addition, we plan to investigate on other pattern languages such as dynamic or attributed graphs, or even 3D graphs. Once again, it requires to adapt the steps of MCTS4DM, and especially the EXPAND and ROLLOUT methods that include the refinement operator.

Towards compressed search trees

As pointed out in the experiments, when local optima are located deeper in the search space, it may require a lot of iterations to reach them. This is even more true when dealing with numerical attributes with a large domain (recall that we consider that the domain of a numerical attribute is made of the different values taken by the objects in the data). Many minimal changes are required (and thus several refinements are performed) to reach a small interval. We proposed a refinement operator based on the generators of the equivalence class that ensures that two nodes in a branch do not have the same support. However, in some cases it is not sufficient. The question is how to compress the search tree, i.e., how to reduce the number of generated nodes, without loss of information? A possible solution would be to incorporate expert's knowledge to prune some parts of the search space that are known to be not interesting. However, we would like to keep the method as generic as possible to be able to apply it to different application domains. Taking into account the theory developed over the past twenty years in pattern mining (upper bounds, condensed representations, ...) [74] seems to be the wisest solution to reduce the size of the search tree in the MCTS algorithm. Indeed, several enhancements are defined to reduce the number of generated patterns with or without loss of information.

Multi-objective pattern mining

Up to now, we use one quality measure to evaluate a pattern. However, in some applications, a measure is not enough to quantify the interestingness of a pattern. In pattern mining, the sky-patterns have been defined when dealing with patterns evaluated with several measures [130]. In this setting, no thresholds are used: The aim is to extract only the dominant patterns w.r.t. the several measures using the Pareto frontier. This is a multi-objective optimization task. Using several measures in a MCTS means that the reward of each simulation is no longer a value but a set of values, corresponding to the several measures. Then, it is required to study the upper confidence bound formula when selecting a child of a node. Indeed, how to take into account several measures to compute UCB? The aggregated value of the rewards of a node $Q(.)$ requires to be defined with the several measures: The Pareto frontier may be used for that.

Bibliography

[1] B. Abramson. Expected-outcome: A general model of static evaluation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(2):182–193, 1990.

[2] T. Abudawood and P. A. Flach. Evaluation measures for multi-class subgroup discovery. In *ECML/PKDD, Part I*, pages 35–50, 2009.

[3] C. C. Aggarwal. *Data Mining - The Textbook*. Springer, 2015.

[4] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD*, pages 207–216, 1993.

[5] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.

[6] R. Agrawal and R. Srikant. Mining sequential patterns. In *IEEE ICDE*, pages 3–14, 1995.

[7] S. Andrews. A partial-closure canonicity test to increase the efficiency of cbo-type algorithms. In *ICCS*, pages 37–50, 2014.

[8] S. Andrews. A 'best-of-breed' approach for designing a fast algorithm for computing fixpoints of galois connections. *Inf. Sci.*, 295:633–649, 2015.

[9] S. Arctander. *Perfume and flavor materials of natural origin*, volume 2. Allured Publishing Corp., 1994.

[10] M. Atzmueller and F. Lemmerich. VIKAMINE - open-source subgroup discovery, pattern mining, and analytics. In *ECML/PKDD*, pages 842–845, 2012.

[11] M. Atzmüller and F. Lemmerich. Fast subgroup discovery for continuous target concepts. In *ISMIS*, pages 35–44, 2009.

[12] M. Atzmüller and F. Puppe. SD-map - A fast algorithm for exhaustive subgroup discovery. In *PKDD*, pages 6–17, 2006.

[13] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[14] T. Bäck, D. B. Fogel, and Z. Michalewicz. Handbook of evolutionary computation. *Release*, 97(1):B1, 1997.

[15] S. D. Bay and M. J. Pazzani. Detecting group differences: Mining contrast sets. *Data Min. Knowl. Discov.*, 5(3):213–246, 2001.

[16] A. Belfodil, S. O. Kuznetsov, C. Robardet, and M. Kaytoue. Mining convex polygon patterns with formal concept analysis. In *IJCAI, In Press*, 2017.

[17] A. Bendimerad, M. Plantevit, and C. Robardet. Unsupervised exceptional attributed subgraph mining in urban data. In *IEEE ICDM*, pages 1–12, 2016.

[18] P. Berkhin. A survey of clustering data mining techniques. In *Grouping Multidimensional Data - Recent Advances in Clustering*, pages 25–71. Springer, 2006.

[19] F. J. Berlanga, M. J. del Jesús, P. González, F. Herrera, and M. Mesonero. Multiobjective evolutionary induction of subgroup discovery fuzzy rules: A case study in marketing. In *Advances in Data Mining, Applications in Medicine, Web Mining, Marketing, Image and Signal Mining, co-located with 6th Industrial Conference on Data Mining*, pages 337–349, 2006.

[20] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis. Mining graph evolution rules. In *ECML/PKDD, Part I*, pages 115–130, 2009.

[21] J. Besson, C. Robardet, J. Boulicaut, and S. Rome. Constraint-based concept mining and its application to microarray data analysis. *Intell. Data Anal.*, pages 59–82, 2005.

[22] Y. Björnsson and H. Finnsson. Cadiaplayer: A simulation-based general game player. *IEEE Trans. Comput. Intellig. and AI in Games*, 1(1):4–15, 2009.

[23] M. Boley, C. Lucchese, D. Paurat, and T. Gärtner. Direct local pattern sampling by efficient two-step random procedures. In *ACM SIGKDD*, pages 582–590, 2011.

[24] G. Bosc, J. Boulicaut, C. Raïssi, and M. Kaytoue. Découverte de sous-groupes avec les arbres de recherche de monte carlo. In *EGC*, pages 23–37, 2017.

[25] G. Bosc, J. Golebiowski, M. Bensafi, C. Robardet, M. Plantevit, J. Boulicaut, and M. Kaytoue. Local subgroup discovery for eliciting and understanding new structure-odor relationships. In *DS*, pages 19–34, 2016.

[26] G. Bosc, M. Kaytoue, M. Plantevit, F. D. Marchi, M. Bensafi, and J. Boulicaut. Vers la découverte de modèles exceptionnels locaux : des règles descriptives liant les molécules à leurs odeurs. In *EGC*, pages 305–316, 2015.

[27] G. Bosc, M. Kaytoue-Uberall, C. Raïssi, and J. Boulicaut. Fouille de motifs séquentiels pour l'élicitation de stratégies à partir de traces d'interactions entre agents en compétition. In *EGC*, pages 359–370, 2014.

[28] G. Bosc, M. Kaytoue-Uberall, C. Raïssi, J. Boulicaut, and P. Tan. Mining balanced sequential patterns in RTS games. In *ECAI*, pages 975–976, 2014.

[29] G. Bosc, M. Plantevit, J. Boulicaut, M. Bensafi, and M. Kaytoue. h(odor): Interactive discovery of hypotheses on the structure-odor relationship in neuroscience. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD, Part III*, pages 17–21, 2016.

[30] G. Bosc, C. Raïssi, J. Boulicaut, and M. Kaytoue. Any-time diverse subgroup discovery with monte carlo tree search. *Major revision in Data Min. Knowl. Discov. journal*, 2017. Available at `http://arxiv.org/abs/1609.08827`.

[31] G. Bosc, P. Tan, J. Boulicaut, C. Raïssi, and M. Kaytoue. A pattern mining approach to study strategy balance in RTS games. *IEEE Trans. Comput. Intellig. and AI in Games*, 9(2):123–132, 2017.

[32] J. F. Boulicaut, L. De Raedt, and H. Mannila, editors. *Constraint-Based Mining and Inductive Databases*, volume 3848 of *LNCS*. Springer, 2005.

[33] J. F. Boulicaut and B. Jeudy. Constraint-based data mining. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook, 2nd ed.*, pages 339–354. Springer, 2010.

[34] R. J. Brachman and T. Anand. The process of knowledge discovery in databases. *Advances in Knowledge Discovery and Data Mining*, pages 37–57, 1996.

[35] B. Bringmann and A. Zimmermann. One in a million: picking the right patterns. *Knowl. Inf. Syst.*, 18(1):61–81, 2009.

[36] C. Browne, E. J. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. P. Liebana, S. Samothrakis, and S. Colton. A survey of monte carlo tree search methods. *IEEE Trans. Comput. Intellig. and AI in Games*, 4(1):1–43, 2012.

[37] L. Buck and R. Axel. A novel multigene family may encode odorant receptors: a molecular basis for odor recognition. *Cell*, 65(1):175–187, 1991.

[38] D. Burdick, M. Calimlim, and J. Gehrke. MAFIA: A maximal frequent itemset algorithm for transactional databases. In *ICDE*, pages 443–452, 2001.

[39] T. Calders, C. Rigotti, and J. Boulicaut. A survey on condensed representations for frequent sets. In *Constraint-Based Mining and Inductive Databases, European Workshop on Inductive Databases and Constraint Based Mining, Hinterzarten, Germany, March 11-13, 2004, Revised Selected Papers*, pages 64–80, 2004.

[40] C. J. Carmona, P.González, M. J. del Jesús, and F. Herrera. NMEEF-SD: non-dominated multiobjective evolutionary algorithm for extracting fuzzy rules in subgroup discovery. *IEEE Trans. Fuzzy Systems*, 18(5):958–970, 2010.

[41] J. B. Castro, A. Ramanathan, and C. S. Chennubhotla. Categorical dimensions of human odor descriptor space revealed by non-negative matrix factorization. *PLOS ONE*, 8(9), 09 2013.

[42] T. Cazenave. Generalized rapid action value estimation. In *IJCAI*, pages 754–760, 2015.

[43] T. Cazenave and N. Jouandeau. On the parallelization of uct. In *proceedings of the Computer Games Workshop*, pages 93–101, 2007.

[44] L. Cerf, J. Besson, K. Nguyen, and J. Boulicaut. Closed and noise-tolerant patterns in n-ary relations. *Data Min. Knowl. Discov.*, 26(3):574–619, 2013.

[45] G. Chaslot, M. Winands, J. Uiterwijk, H. Van Den Herik, and B. Bouzy. Progressive strategies for monte-carlo tree search. In *JCIS*, pages 655–661, 2007.

[46] G. Chaslot, M. H. M. Winands, and H. J. van den Herik. Parallel monte-carlo tree search. In *CG*, pages 60–71, 2008.

[47] V. Codocedo, G. Bosc, M. Kaytoue, J. Boulicaut, and A. Napoli. A proposition for sequence mining using pattern structures. In *ICFCA*, pages 106–121, 2017.

[48] D. J. Cook and L. B. Holder. *Mining graph data*. John Wiley & Sons, 2006.

[49] R. Coulom. Computing "elo ratings" of move patterns in the game of go. *ICGA Journal*, 30(4):198–208, 2007.

[50] C. A. de March, S. Ryu, G. Sicard, C. Moon, and J. Golebiowski. Structure-odour relationships reviewed in the postgenomic era. *Flavour and Fragrance Journal*, 30(5):342–361, 2015.

[51] L. De Raedt, M. Jaeger, S. D. Lee, and H. Mannila. A theory of inductive query answering. In *IEEE ICDM*, pages 123–130, 2002.

[52] C. R. de Sá, W. Duivesteijn, C. Soares, and A. J. Knobbe. Exceptional preferences mining. In *DS*, pages 3–18, 2016.

[53] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation*, 6(2):182–197, 2002.

[54] M. J. del Jesús, P. González, F. Herrera, and M. Mesonero. Evolutionary fuzzy rule induction process for subgroup discovery: A case study in marketing. *IEEE Trans. Fuzzy Systems*, 15(4):578–592, 2007.

[55] C. Delasalle, C. A. de March, U. J. Meierhenrich, H. Brevard, J. Golebiowski, and N. Baldovini. Structure-odor relationships of semisynthetic β-santalol analogs. *Chemistry & Biodiversity*, 11(11):1843–1860, 2014.

[56] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *ACM SIGKDD*, pages 43–52, 1999.

[57] L. Downar and W. Duivesteijn. Exceptionally monotone models - the rank correlation model class for exceptional model mining. In *IEEE ICDM*, pages 111–120, 2015.

[58] W. Duivesteijn, A. Feelders, and A. J. Knobbe. Different slopes for different folks: mining for exceptional regression models with cook's distance. In *ACM SIGKDD*, pages 868–876, 2012.

[59] W. Duivesteijn, A. Feelders, and A. J. Knobbe. Exceptional model mining - supervised descriptive local pattern mining with complex target concepts. *Data Min. Knowl. Discov.*, 30(1):47–98, 2016.

[60] W. Duivesteijn and A. J. Knobbe. Exploiting false discoveries - statistical validation of patterns and quality measures in subgroup discovery. In *IEEE ICDM*, pages 151–160, 2011.

[61] W. Duivesteijn, A. J. Knobbe, A. Feelders, and M. van Leeuwen. Subgroup discovery meets bayesian networks – an exceptional model mining approach. In *IEEE ICDM*, pages 158–167, 2010.

[62] S. Dzeroski, B. Goethals, and P. Panov, editors. *Inductive Databases and Constraint-Based Data Mining*. Springer, 2010.

[63] V. Dzyuba and M. van Leeuwen. Interactive discovery of interesting subgroup sets. In *IDA*, pages 150–161, 2013.

[64] V. Dzyuba, M. van Leeuwen, S. Nijssen, and L. De Raedt. Interactive learning of pattern rankings. *International Journal on Artificial Intelligence Tools*, 23(6), 2014.

[65] A. K. et al. Dream olfaction prediction challenge, 2015. Sponsors: IFF, IBM Research, Sage Bionetworks and DREAM. URL: `www.synapse.org/#!Synapse:syn2811262`.

[66] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *IJCAI*, pages 1022–1029, 1993.

[67] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. *Advances in Knowledge Discovery and Data Mining*, pages 1–34, 1996.

[68] J. Fürnkranz, D. Gamberger, and N. Lavrac. *Foundations of Rule Learning*. Cognitive Technologies. Springer, 2012.

[69] D. Gamberger and N. Lavrac. Expert-guided subgroup discovery: Methodology and application. *J. Artif. Intell. Res.*, 17:501–527, 2002.

[70] B. Ganter and R. Wille. *Formal concept analysis - mathematical foundations*. Springer, 1999.

[71] R. Gaudel and M. Sebag. Feature selection as a one-player game. In *IEEE ICDM*, pages 359–366, 2010.

[72] S. Gelly. *A contribution to Reinforcement Learning; application to Computer-Go*. PhD thesis, Université Paris-Sud, 2007.

[73] S. Gelly and D. Silver. Combining online and offline knowledge in UCT. In *ICML 2007*, pages 273–280, 2007.

[74] A. Giacometti, D. H. Li, P. Marcel, and A. Soulet. 20 years of pattern mining: a bibliometric survey. *SIGKDD Explorations*, 15(1):41–50, 2013.

[75] K. Gouda and M. J. Zaki. Efficiently mining maximal frequent itemsets. In *IEEE ICDM*, pages 163–170, 2001.

[76] H. Grosskreutz and S. Rüping. On subgroup discovery in numerical domains. *Data Min. Knowl. Discov.*, 19(2):210–226, 2009.

[77] H. Grosskreutz, S. Rüping, and S. Wrobel. Tight optimistic estimates for fast subgroup discovery. In *ECML/PKDD*, pages 440–456, 2008.

[78] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[79] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. *Data Min. Knowl. Discov.*, 15(1):55–86, 2007.

[80] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM SIGMOD*, pages 1–12, 2000.

[81] M. A. Hearst. Trends & controversies: Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28, 1998.

[82] D. P. Helmbold and A. Parker-Wood. All-moves-as-first heuristics in monte-carlo go. In *ICAI*, pages 605–610, 2009.

[83] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press, 1975.

[84] B. Jeudy and J. Boulicaut. Optimization of association rule mining queries. *Intell. Data Anal.*, pages 341–357, 2002.

[85] P. Joussain, A. Chakirian, F. Kermen, C. Rouby, and M. Bensafi. Physicochemical influence on odor hedonics: Where does it occur first? *Communicative & integrative biology*, 4(5):563–565, 2011.

[86] K. Kaeppler and F. Mueller. Odor classification: a review of factors influencing perception-based odor arrangements. *Chemical senses*, 38(3):189–209, 2013.

[87] B. Kavsek, N. Lavrac, and V. Jovanoski. APRIORI-SD: adapting association rule learning to subgroup discovery. In *IDA*, pages 230–241, 2003.

[88] M. Kaytoue, S. O. Kuznetsov, and A. Napoli. Revisiting numerical pattern mining with formal concept analysis. In *IJCAI*, pages 1342–1347, 2011.

[89] M. Kaytoue, S. O. Kuznetsov, A. Napoli, and S. Duplessis. Mining gene expression data with pattern structures in formal concept analysis. *Inf. Sci.*, 181(10):1989–2001, 2011.

[90] M. Kaytoue, M. Plantevit, A. Zimmermann, A. Bendimerad, and C. Robardet. Exceptional contextual subgraph mining. *Machine Learning*, pages 1–41, 2017.

[91] A. Keller, R. C. Gerkin, Y. Guan, A. Dhurandhar, G. Turu, B. Szalai, J. D. Mainland, Y. Ihara, C. W. Yu, R. Wolfinger, C. Vens, L. Schietgat, K. De Grave, R. Norel, D. O. P. Consortium, G. Stolovitzky, G. A. Cecchi, L. B. Vosshall, and P. Meyer. Predicting human olfactory perception from chemical features of odor molecules. *Science*, 355(6327):820–826, 2017.

[92] R. M. Khan, C.-H. Luk, A. Flinker, A. Aggarwal, H. Lapid, R. Haddad, and N. Sobel. Predicting odor pleasantness from odorant structure: pleasantness as a reflection of the physical world. *The Journal of Neuroscience*, 27(37):10015–10023, 2007.

[93] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. ASAI, 1996.

[94] W. Klösgen and M. May. Census data mining, an application. In *PKDD*, pages 65–79, 2002.

[95] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *ECML*, pages 282–293, 2006.

[96] R. M. Konijn, W. Duivesteijn, M. Meeng, and A. J. Knobbe. Cost-based quality measures in subgroup discovery. *J. Intell. Inf. Syst.*, 45(3):337–355, 2015.

[97] P. Krajca, J. Outrata, and V. Vychodil. Advances in algorithms based on CbO. In *CLA*, pages 325–337, 2010.

[98] K. L. Kroeker. A new benchmark for artificial intelligence. *Commun. ACM*, 54(8):13–15, 2011.

[99] S. O. Kuznetsov. A fast algorithm for computing all intersections of objects from an arbitrary semilattice. *Nauchno-Tekhnicheskaya Informatsiya Seriya 2 - Informatsionnye protsessy i sistemy*, 2(1):17–20, 1993.

[100] S. O. Kuznetsov. A fast algorithm for computing all intersections of objects in a finite semilattice. *Automatic Documentation and Mathematical Linguistics*, 27(5):400–412, 1993.

[101] N. Lavrac, B. Cestnik, D. Gamberger, and P. A. Flach. Decision support through subgroup discovery: Three case studies and the lessons learned. *Machine Learning*, 57(1-2):115–143, 2004.

[102] N. Lavrac, P. A. Flach, and B. Zupan. Rule evaluation measures: A unifying view. In *ILP*, pages 174–185, 1999.

[103] Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[104] D. Leman, A. Feelders, and A. J. Knobbe. Exceptional model mining. In *ECML/PKDD*, pages 1–16, 2008.

[105] F. Lemmerich, M. Atzmueller, and F. Puppe. Fast exhaustive subgroup discovery with numerical target concepts. *Data Min. Knowl. Discov.*, 30(3):711–762, 2016.

[106] F. Lemmerich, M. Becker, and M. Atzmueller. Generic pattern trees for exhaustive exceptional model mining. In *ECML PKDD*, pages 277–292, 2012.

[107] F. Lemmerich, M. Rohlfs, and M. Atzmüller. Fast discovery of relevant subgroup patterns. In *FLAIRS*, pages 428–433, 2010.

[108] B. T. Lowerre. *The HARPY speech recognition system.* PhD thesis, Carnegie-Mellon Univ., Pittsburgh, PA. Dept. of Computer Science., 1976.

[109] E. Loza Mencía and F. Janssen. Learning rules for multi-label classification: a stacking and a separate-and-conquer approach. *Machine Learning*, 105(1):77–126, 2016.

[110] J. M. Luna, J. R. Romero, C. Romero, and S. Ventura. Discovering subgroups by means of genetic programming. In *EuroGP*, pages 121–132, 2013.

[111] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Min. Knowl. Discov.*, 1(3):241–258, 1997.

[112] M. Meeng, W. Duivesteijn, and A. J. Knobbe. Rocsearch - an roc-guided search strategy for subgroup discovery. In *IEEE ICDM*, pages 704–712, 2014.

[113] U. J. Meierhenrich, J. Golebiowski, X. Fernandez, and D. Cabrol-Bass. The molecular basis of olfactory chemoreception. *Angewandte Chemie International Edition*, 43(47):6410–6412, 2004.

[114] S. Moens and M. Boley. Instant exceptional model mining using weighted controlled pattern sampling. In *IDA*, pages 203–214, 2014.

[115] S. Morishita and J. Sese. Traversing itemset lattice with statistical metric pruning. In *ACM PODS*, pages 226–236, 2000.

[116] F. Moser, R. Colak, A. Rafiey, and M. Ester. Mining cohesive patterns from graphs with feature vectors. In *SIAM SDM*, pages 593–604, 2009.

[117] M. Mueller, R. Rosales, H. Steck, S. Krishnan, B. Rao, and S. Kramer. Subgroup discovery for test selection: A novel approach and its application to breast cancer diagnosis. In *IDA*, pages 119–130, 2009.

[118] S. Nijssen and A. Zimmermann. Constraint-based pattern mining. In *Frequent Pattern Mining*, pages 147–163. Springer, 2014.

[119] P. K. Novak, N. Lavrac, and G. I. Webb. Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *Journal of Machine Learning Research*, 10:377–403, 2009.

[120] V. Pachón, J. M. Vázquez, J. L. Domínguez, and M. J. M. López. Multi-objective evolutionary approach for subgroup discovery. In *Hybrid Artificial Intelligent Systems*, pages 271–278, 2011.

[121] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Inf. Syst.*, 24(1):25–46, 1999.

[122] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns by prefix-projected growth. In *IEEE ICDE*, pages 215–224, 2001.

[123] N. Ramakrishnan, D. Kumar, B. Mishra, M. Potts, and R. F. Helm. Turning cartwheels: an alternating algorithm for mining redescriptions. In *ACM KDD*, pages 266–275, 2004.

[124] C. Robardet. Constraint-based pattern mining in dynamic graphs. In *IEEE ICDM*, pages 950–955, 2009.

[125] D. Rodríguez, R. Ruiz, J. C. Riquelme, and J. S. Aguilar-Ruiz. Searching for rules to detect defective modules: A subgroup discovery approach. *Inf. Sci.*, 191:14–30, 2012.

[126] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

[127] S. R. Safavian and D. A. Landgrebe. A survey of decision tree classifier methodology. *IEEE Trans. Systems, Man, and Cybernetics*, 21(3):660–674, 1991.

[128] M. P. D. Schadd, M. H. M. Winands, H. J. van den Herik, G. Chaslot, and J. W. H. M. Uiterwijk. Single-player monte-carlo tree search. In *CG*, volume 5131 of *LNCS*, pages 1–12. Springer, 2008.

[129] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and

D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[130] A. Soulet, C. Raïssi, M. Plantevit, and B. Crémilleux. Mining dominant patterns in the sky. In *IEEE ICDM*, pages 655–664, 2011.

[131] G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Computing iceberg concept lattices with Titanic. *Data Knowl. Eng.*, 42(2):189–222, 2002.

[132] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[133] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[134] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook, 2nd ed.*, pages 667–685. Springer, 2010.

[135] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.

[136] D. van der Merwe, S. A. Obiedkov, and D. G. Kourie. Addintent: A new incremental algorithm for constructing concept lattices. In *ICFCA*, pages 372–385, 2004.

[137] M. van Leeuwen and E. Galbrun. Association discovery in two-view data. *IEEE Trans. Knowl. Data Eng.*, 27(12):3190–3202, 2015.

[138] M. van Leeuwen and A. J. Knobbe. Non-redundant subgroup discovery in large and complex data. In *ECML/PKDD*, pages 459–474, 2011.

[139] M. van Leeuwen and A. J. Knobbe. Diverse subgroup set discovery. *Data Min. Knowl. Discov.*, 25(2):208–242, 2012.

[140] M. van Leeuwen and A. Ukkonen. Discovering skylines of subgroup sets. In *ECML/PKDD*, pages 272–287, 2013.

[141] I. H. Witten, F. Eibe, and M. A. Hall. *Data mining: practical machine learning tools and techniques, 3rd Edition*. Morgan Kaufmann, Elsevier, 2011.

[142] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *PKDD*, pages 78–87, 1997.

[143] M. J. Zaki and C. Hsiao. CHARM: an efficient algorithm for closed itemset mining. In *SIAM SDM*, pages 457–473, 2002.

[144] M. J. Zaki and W. M. Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, 2014.

[145] E. Zitzler, M. Laumanns, and L. e. a. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. In *Eurogen*, volume 3242, pages 95–100, 2001.

Résumé

La découverte de motifs qui caractérisent fortement une classe vis à vis d'une autre reste encore un problème difficile en fouille de données. La découverte de sous-groupes (Subgroup Discovery, SD) est une approche formelle de fouille de motifs qui permet la construction de classifieurs intelligibles mais surtout d'émettre des hypothèses sur les données. Cependant, cette approche fait encore face à deux problèmes majeurs : (i) comment définir des mesures de qualité appropriées pour caractériser l'intérêt d'un motif et (ii) comment sélectionner une méthode heuristique adaptée lorsqu'une énumération exhaustive de l'espace de recherche n'est pas réalisable. Le premier problème a été résolu par la fouille de modèles exceptionnels (Exceptional Model Mining, EMM) qui permet l'extraction de motifs couvrant des objets de la base de données pour lesquels le modèle induit sur les attributs de classe est significativement différent du modèle induit par l'ensemble des objets du jeu de données. Le second problème a été étudié en SD et EMM principalement avec la mise en place de méthodes heuristiques de type recherche en faisceau (beam-search) ou avec des algorithmes génétiques qui permettent la découverte de motifs non redondants, diversifiés et de bonne qualité. Dans cette thèse, nous soutenons que la nature gloutonne des méthodes d'énumération précédentes génère cependant des ensembles de motifs manquant de diversité. Nous définissons formellement la fouille de données comme un jeu que nous résolvons par l'utilisation de la recherche arborescente de Monte Carlo (Monte Carlo Tree Search, MCTS), une technique récente principalement utilisée pour la résolution de jeux et de problèmes de planning en intelligence artificielle. Contrairement aux méthodes traditionnelles d'échantillonnage, MCTS donne la possibilité d'obtenir une solution à tout instant sans qu'aucune hypothèse ne soit faite que ce soit sur la mesure de qualité ou sur les données. Cette méthode d'énumération converge vers une approche exhaustive si les budgets temps et mémoire disponibles sont suffisants. Le compromis entre l'exploration et l'exploitation que propose cette approche permet une augmentation significative de la diversité dans l'ensemble des motifs calculés. Nous montrons que la recherche arborescente de Monte Carlo appliquée à la fouille de motifs permet de trouver rapidement un ensemble de motifs diversifiés et de bonne qualité à l'aide d'expérimentations sur des jeux de données de référence et sur un jeu de données réel traitant de l'olfaction. Nous proposons et validons également une nouvelle mesure de qualité spécialement conçue pour des jeux de donnée multi labels présentant une grande variance de fréquences des labels.

Mots-clés: Découverte de connaissances, règles supervisées, sous-groupes, fouille de modèles exceptionnels, recherche arborescente de Monte Carlo, diversité, olfaction

Abstract

The discovery of patterns that strongly distinguish one class label from another is still a challenging data-mining task. Subgroup Discovery (SD) is a formal pattern mining framework that enables the construction of intelligible classifiers, and, most importantly, to elicit interesting hypotheses from the data. However, SD still faces two major issues: (i) how to define appropriate quality measures to characterize the interestingness of a pattern; (ii) how to select an accurate heuristic search technique when exhaustive enumeration of the pattern space is unfeasible. The first issue has been tackled by Exceptional Model Mining (EMM) for discovering patterns that

cover tuples that locally induce a model substantially different from the model of the whole dataset. The second issue has been studied in SD and EMM mainly with the use of beam-search strategies and genetic algorithms for discovering a pattern set that is non-redundant, diverse and of high quality. In this thesis, we argue that the greedy nature of most such previous approaches produces pattern sets that lack diversity. Consequently, we formally define pattern mining as a game and solve it with Monte Carlo Tree Search (MCTS), a recent technique mainly used for games and planning problems in artificial intelligence. Contrary to traditional sampling methods, MCTS leads to an any-time pattern mining approach without assumptions on either the quality measure or the data. It converges to an exhaustive search if given enough time and memory. The exploration/exploitation trade-off allows the diversity of the result set to be improved considerably compared to existing heuristics. We show that MCTS quickly finds a diverse pattern set of high quality in our application in neurosciences. We also propose and validate a new quality measure especially tuned for imbalanced multi-label data.

Keywords: Knowledge discovery in databases, supervised rules discovery, subgroup discovery, exceptional model mining, monte carlo tree search, diversity, olfaction

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : BOSC
(avec précision du nom de jeune fille, le cas échéant)

DATE de SOUTENANCE : 11/09/2017

Prénoms : Guillaume

TITRE : Anytime discovery of a diverse set of patterns with Monte Carlo tree search

NATURE : Doctorat

Numéro d'ordre : AAAALYSEIXXXX

Ecole doctorale : InfoMaths (ED 512)

Spécialité : Informatique

RESUME :
La découverte de motifs qui caractérisent fortement une classe vis à vis d'une autre reste encore un problème difficile en fouille de données. La découverte de sous-groupes (Subgroup Discovery, SD) est une approche formelle de fouille de motifs qui permet la construction de classifieurs intelligibles mais surtout d'émettre des hypothèses sur les données. Cependant, cette approche fait encore face à deux problèmes majeurs : (i) comment définir des mesures de qualité appropriées pour caractériser l'intérêt d'un motif et (ii) comment sélectionner une méthode heuristique adaptée lorsqu'une énumération exhaustive de l'espace de recherche n'est pas réalisable. Le premier problème a été résolu par la fouille de modèles exceptionnels (Exceptional Model Mining, EMM) qui permet l'extraction de motifs couvrant des objets de la base de données pour lesquels le modèle induit sur les attributs de classe est significativement différent du modèle induit par l'ensemble des objets du jeu de données. Le second problème a été étudié en SD et EMM principalement avec la mise en place de méthodes heuristiques de type recherche en faisceau (beam-search) ou avec des algorithmes génétiques qui permettent la découverte de motifs non redondants, diversifiés et de bonne qualité. Dans cette thèse, nous soutenons que la nature gloutonne des méthodes d'énumération précédentes génère cependant des ensembles de motifs manquant de diversité.
Nous définissons formellement la fouille de données comme un jeu que nous résolvons par l'utilisation de la recherche arborescente de Monte Carlo (Monte Carlo Tree Search, MCTS), une technique récente principalement utilisée pour la résolution de jeux et de problèmes de planning en intelligence artificielle. Contrairement aux méthodes traditionnelles d'échantillonnage, MCTS donne la possibilité d'obtenir une solution à tout instant sans qu'aucune hypothèse ne soit faite que ce soit sur la mesure de qualité ou sur les données. Cette méthode d'énumération converge vers une approche exhaustive si les budgets temps et mémoire disponibles sont suffisants. Le compromis entre l'exploration et l'exploitation que propose cette approche permet une augmentation significative de la diversité dans l'ensemble des motifs calculés. Nous montrons que la recherche arborescente de Monte Carlo appliquée à la fouille de motifs permet de trouver rapidement un ensemble de motifs diversifiés et de bonne qualité à l'aide d'expérimentations sur des jeux de données de référence et sur un jeu de données réel traitant de l'olfaction. Nous proposons et validons également une nouvelle mesure de qualité spécialement conçue pour des jeux de donnée multi labels présentant une grande variance de fréquences des labels.

MOTS-CLÉS : Découverte de connaissances, règles supervisées, sous-groupes, fouille de modèles exceptionnels, recherche arborescente de Monte Carlo, diversité, olfaction

Laboratoire (s) de recherche : Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS)

Directeur de thèse:
Jean-François Boulicaut (Professeur des Universités, INSA de Lyon),
Mehdi Kaytoue (Maître de Conférences, INSA de Lyon)

Président de jury :

Composition du jury :
Toon Calders (Professeur des Universités, Univesiteit Antwerpen)
Tristan Cazenave (Professeur des Universités, Université Paris-Dauphine)
Sihem Amer-Yahia (Directrice de recherche, CNRS)
Moustafa Bensafi (Directeur de Recherche, CNRS)
Peter Flach (Professeur des Universités, University of Bristol)
Katharina Morik (Professeure des Universités, Technische Universität Dortmund)