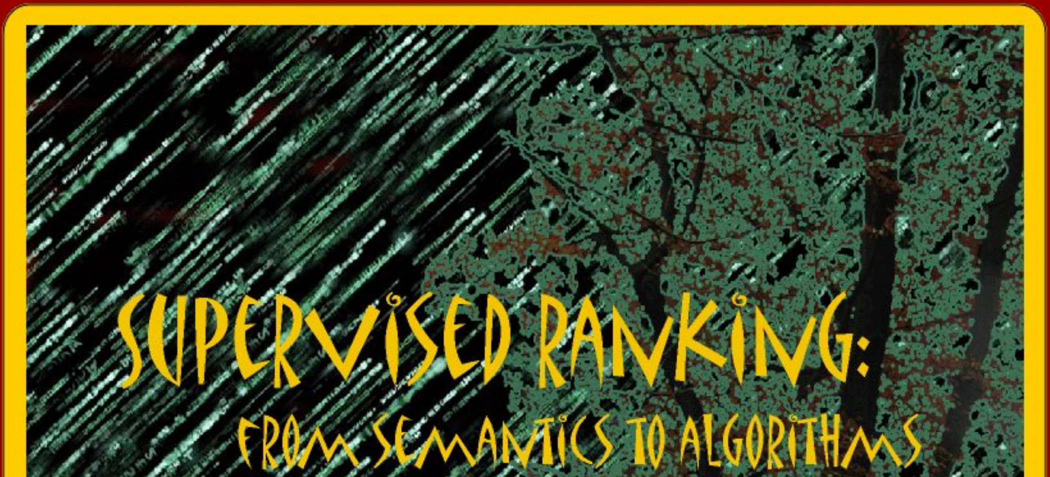


SUPERVISED RANKING:
FROM SEMANTICS TO ALGORITHMS

KIM CAO-VAN

UNIVERSITEIT GENT

Faculty of Sciences

Supervised ranking

from semantics to algorithms

Kim Cao-Van

November 2003
Dissertation in fulfillment of the requirements for the degree of
Doctor of Philosophy (Ph.D.)

Promotor: *prof. dr. Bernard De Baets*

Onderzoekseenheid	Research Unit
“Kennisgebaseerde Systemen”	*“Knowledge-based Systems”*
Vakgroep Toegepaste Wiskunde,	Department of Applied Mathematics,
Biometrie en Procesregeling	Biometrics and Process Control
Universiteit Gent	Ghent University
Coupure links 653	Coupure links 653
9000 Gent	B-9000 Ghent
België	Belgium

CONTACT VIA EMAIL:

`kim@biomath.UGent.be (kim_caovan@hotmail.com)`
`Bernard.DeBaets@UGent.be`

Deze thesis werd financieel ondersteund door Universiteit Gent.
This thesis was financially supported by Ghent University.
Bijzonder Onderzoeksfonds (BOF) 011D5399.

Aan mijn vader,
to my father,
kính tặng cha tôi.

Prelude

Time to say thank you to everybody who helped me achieve this incredible feat. So, thank you everybody!

No, of course, I'm just kidding. You people deserve far more than such a simplistic short and immensely impersonal phlegmatic phrase. But where to start?

Well, why not with the beginning? That means my dearest parents, the first people I ever laid eyes upon and still my biggest support. I couldn't have wished for better parents than them, and, in fact, this thesis is a nice reflection of them both: my father, the mathematician, with his really analytical and logical mind, and my mother, the psychologist and human resources manager, with her holistic and intuitive mind. So it seems that the style of this thesis was predestined. The next person I saw in my life was my sister, the greatest sis in the whole world, although a bit peculiar and special at times (well, let's be honest, all the time). Although she told me that her very first words as a 3-year old to me as a baby were not very nice (to say the least), now, 27 years later, we are connected as if born as twins. Without such a great family, I couldn't have brought this to an end.

And then, of course, Bernard. A really great mentor. I was quite spoiled in the beginning being his first and sole Ph.D. student. As a matter of fact, I took up my term the same day he assumed his position as a professor. Truly a superb start for a superb collaboration: two creative minds together, both a bit out-of-this-world, but mine kept somewhat in measure by the experience of Bernard. He knew how to stimulate my interest by offering me rather arduous and audacious challenges (the ultimate way to get me to work: he actually succeeded in making me write my first article within the first one month and a half). At the same time, he knew instinctively when to grant me some more quiet moments, and to let me liberate myself from all the rigmarole inextricably bound up with research. So, thank you... thank you very much.

What would I have been without my many precious and priceless friends? They offered me so much support, directly with a helping hand, a listening ear and whole-hearted advice – sometimes a bit useless, but hey, it's the though that counts – while I overwhelmed them with all my blissful happiness and endless sorrow, euphoria and frustration, in short, with all the mood swings I was exposed to during the writing of this Ph.D (you'll hear all and more about those in the Interludes dispersed throughout this booklet). Indirectly by just being there, by not complaining when I neglected and – let's be honest – altogether forget them during some long stressful periods bursting with deadlines. Or by brightening up all the short and longer climbing trips (I cannot help bringing up the sweet memory of the superb trip to the Ardeche – O dddddear dear – and all those days in Fontainebleau). I am not going to sum up everybody, that's just too boring to read. Anyway, they know who they are and most of them will be present at my defense, and, of course... the after-party!

Gent, November 2003

Contents

Interlude

WRITER'S RUSH

FRIDAY, MAY 16, 2003. *Well people, this is it! Time for the final rush. Yesterday evening, I went gallivanting with Ruben, ending up in the pub (the* "Hotsy Totsy"*), and we were babbling about this and that, and at a certain moment I said:* "I have 6 weeks left, and about 9 chapters, that makes I have to finish, euh... one and a half chapter a week." *That was the moment it first struck me, the truth I had kept blocking from myself came swirling and trusting upon me with a fiercer intensity than I could have hardly imagined: time was definitely not on my side.*
Now, back at home, I have just jotted down the general layout and found 10 chapters, rather than 9, making matters even worse. But a spark of light soothes my mind: 2 of them are already close to being completed. On the other hand, the remaining 8 are still frightening blank. Certainly chapters 6 and 7, because I had to rethink most of their contents due to some nasty little (sic) problems I but discovered in April (see the Interlude on page 175). I guess this marks the beginning of a forced reclusive period for me.

WEDNESDAY, MAY 28. *Only 4 weeks to go, and still 7 chapters left (I just completed Chapter 6). This night, I had a bad dream. It started out as a very pleasant dream at first, I was really having fun, until at a certain moment, someone started to sing "Happy birthday..." to somebody. Did I forget a birthday?* Yes, *the answer came,* it's the 13th of June today. *No way.* Yes, it is. *No, it can't be, I know it was yesterday the 28th of May*[1], *how can two weeks simply disappear?* They didn't. *But how... And I could see there annoyed faces, as if saying, hey, stop acting like a fool. Panick crept onto me, how will I ever finish writing in time? And then I woke up, and realised it was but a dream.*
So, here I am again, frittering away my precious time writing a nonsensical interlude. It's all about setting priorities ;-) and I do draw strengt and courage from them. Well, enough of this, let's tackle Chapter 7!

SATURDAY, JUNE 14. *No bad dreams anymore, I didn't forget the birthday and I didn't loose two weeks. (Life is great, isn't it?) Chapter 7 has been wrapped up and is ready for review (well, except for the experiments and some minor stuff), and I'm finishing Chapter 5. Only 2 weeks to go and assuming number 5 will be concluded tomorrow – clearly an example of wishful thinking – still 6 chapters left . Hey?? $7-2=6$?? Aha, it seems that last time, I forgot to count in the "weka" chapter Bernard once asked me to*

[1]Although it was a dream, this makes perfectly sense because I worked late yesterday night, so I saw the date May 28 just before going to bed.

write. Oh well, what is one chapter more or less... It's 2:25 am, I guess I'm simply to tired to care. Time to drag my sleepy slumbering self to my snug and snoozy bed.

THURSDAY, JUNE 19. *As expected, wishful thinking. I finished Chapter 5 only just now. Maybe it's time for me to admit my deadline of June 30 is not really feasible. Although, not yet... Let's stay surrealistically optimistic and foolishly confident in the positive outcome, against all down to earth rationality. I already get enough of that in my thesis. So, I'll just keep fooling myself a little longer, and to make this dreamlike thought even more complete, I'll take the evening off to go to a puppet theater play (a mix of puppets and actors) in Antwerp.*

FRIDAY, JUNE 20. *The play (*"Love in Babylon"* by Froefroe) was superb, I'm all excited about it and feel a sudden surge of thrilling inspiration for the play I intend to create next year. But first things first, I still have to conclude some unfinished business, like this thesis ;-)*

WEDNESDAY, JUNE 25. *I just entrusted the last words of Chapter 3 to electronic paper! 5 days left, the final ultimate rush, and how many chapters left? No time to count, work, work, work!!!*

TUESDAY, JULY 1. *Well, this is it. Unbelievable but true, I made it. Although I have to admit with a silent sideways whisper that I kind of tricked and cheated my way to the end: I condensed two chapters into shorter appendices ($5-2=3$), and I completely erased what where meant to become the two final chapters ($3-2=1$), about a measure for rankings and about the weka implementations. Anyway, in spite of the year of thinking and fretting about the ranking measure, I am still not fully satisfied about it, so I prefer not to include it. And the other chapter is not even essential to this thesis (moreover, I was never really keen on putting it in). Also, I still need to put together the first chapter (the appendix is finished though), but that's "just" – to put my conscience at ease – the introductory chapter.*

Chapter 1

Philosophy and problem setting

1.1 Introduction

Telling a story is an art in itself. To each story, there are a multitude of possible viewing angles, each with their own characteristic shades, colours and accents. Each tale has its share of characters and apt sceneries blending one into the other, consists of different narrative threads skilfully interwoven to create some grand tapestry that cannot be conceived of by simply laying these threads next to each other. Each tale brings forth an accumulation of plots and twists and turns wired together like beads on a string, ever tickling the imagination, ever leading to an even larger and more intricate plot. And is a Ph.D. dissertation, this scientific concoction of thoughts, ideas, intuition and insight, not just a story? Characters become notions, sceneries transform into frameworks, and all are interconnected and influencing each other. And what else is the constructive working towards a result, if not some plot unravelling?

Since I will be writing a story, there is no need to heed the chronological evolution of the research I so painstakingly conducted during the past four years. Rather, I will focus on following the natural flow, starting from elementary basics about perception (semantics) and representation (syntax), moulding them into rudimentary but solid building blocks (the framework) for our edifice, and gradually whittling them into a finer shape (the algorithms) whilst continuing to pile and stack them one upon the other. But in spite of this underlying coherent current so conspicuously carving through the consecutive chapters, I strived to make each of these units as self-contained as possible, standing alone in their connectedness as the islets and islands of an archipel.

But before commencing this tale, I will unfold in this prologue chapter the basic concepts and philosophy upon which all my research is founded. It is the source of the river meandering through this thesis, the force sweeping away all obstacles and leaving in its wake a fertile ground from which ideas can emerge. It feeds the bare, desolate and fallow plains of unsolved problems, fostering them into rich arable fields and forests dense with foliage.

1.2 Problem setting

The ultimate trigger to this thesis can be easily traced back to a bunch of articles and reports by the hand of Greco, Słowiński and Mattarazo, proposing a method to extend the principles behind rough set analysis from classification towards ranking. Plunging into these papers, the question popped up whether a similar feat was feasible for decision trees (since the first time I encountered these tree structures during my studies, I never really lost my fondness of them). Little did I know such extensions were already proposed in the literature during the last years. Maybe this was but a fortunate trick of fate, since it allowed me to develop my own philosophy and theories without any interfering thoughts gnawing at the back of my mind.

The words *ranking, classification, rough set analysis* and *decision trees* have been spilled out on stage, so it is probably worth while to commence with a little background to enlighten these concepts.

1.2.1 Supervised ranking

What is supervised learning? Such a question is easily solved by grabbing some explanatory dictionary and simply look it up. However, because it is not an entry in the dictionary, we fall back on an encyclopedia:

> **supervised learning.** *Machine learning:* Creating a function from training data. The training data consists of pairs of input objects (typically vectors), and desired outputs. The output of the function can be a continuous value (called regression), or can predict a class label of the input object (called classification). The task of the supervised learner is to predict the value of the function for any valid input object after having seen only a small number of training examples (i.e. pairs of input and target output). To achieve this, the learner has to generalise from the presented data to unseen situations in a "reasonable" way. – *Wikipedia, the free encyclopedia.*

The *supervised learner* is some computer algorithm, like decision trees (see Appendix 1.A), neural nets, nearest neighbours, methods based on the rough set approach (see Appendix 1.B), and many others... Most of these techniques exist both in a *classification*-format and in *regression*-format. A small example of training data and a *classifier* (in this case a classification tree) induced from this data is given in Figure 1.1.

(a) An example of training data.

	c_1	c_2	c_3	class label
a_1	$-$	$-$	$+$	A
a_2	$+$	$-$	$-$	B
a_3	$-$	$+$	$+$	C
a_4	$+$	$+$	$-$	B
object id	input vectors			desired output

(b) Classification tree.

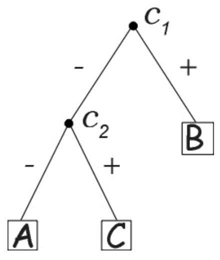


Figure 1.1: An example of supervised learning.

What is ranking? This time, we are more lucky with consulting our dictionary, finding as an explanation:

> **rank.** *Noun:* A relative position or degree of value in a graded group. *Transitive verb:* 1. To place in a row or rows. 2. To give a particular order or position to; classify. 3. To outrank or take precedence over. – *The American Heritage® Dictionary of the English Language, Fourth Edition.*

We learn from this definition that *ranking* is a special form of *classification*, where the concepts of *order* and *relative* position play an important role. Moreover, it is strongly linked with the idea of *outranking*.

Putting it together. Melting these two concepts together is easy enough to understand. Now the class label to be predicted is a relative position in a graded group, where the grade of a group indicates which of the other groups are (or are not) preferred over it. So instead of class labels like “Red”, “Blue” and “Yellow”, we have rank labels like “Bad”, “Moderate” and “Good”.
From a theoretical perspective, we may suspect that we will have to endeavour the mingling of machine learning techniques and statistics with order theory and preference modelling.

1.2.2 Problems of a monotone nature

> **mon·o·tone.** *Noun:* Sameness or dull repetition in sound, style, manner, or colour. *Adjective:* 1. Characterised by or uttered in a monotone. 2. also **mon·o·ton·ic** *Mathematics:* Designating sequences, the successive members of which either consistently increase or decrease but do not oscillate in relative value. Each member of a monotone increasing sequence is greater than or equal to the preceding member; each member of a monotone decreasing sequence is less than or equal to the preceding member. *– The American Heritage® Dictionary of the English Language, Fourth Edition.*

Rankings differ from classifications in more than just the order on the label. As can be seen from the above definitions, there is also a semantical component involved: a higher ranked object is preferred over one with a lower rank. This characteristic proper to rankings may lead to the following problem regarding monotonicity (in the mathematical sense):

Example. Assume four candidates are applying for a job. They are evaluated according to their working experience (little or much), their capacity for learning (slow or fast), and their personal profile, i.e. how well they will fit into the group they have to work with (bad or good). These criteria are denoted resp. by c_1, c_2 and c_3, and their values are denoted by $-$ and $+$. Finally, some committee gives the candidates a global evaluation (B(ad), M(oderate) or G(ood)). They do this as follows: The first candidate a_1 has little work experience, is a slow learner, but fits perfectly in the group. So, this would be someone who won't be able to do much, and will be tattering and babbling to the others, keeping them too from working. Obviously not the perfect candidate, and (s)he is evaluated as Bad. The second candidate a_2 has ample work experience, but is not a keen learner and not a very good team player. It is decided this candidate is Moderately acceptable. Candidate a_3 has no work experience, but compensates this by being a fast learner, and (s)he has the right personal profile. This seems to be a Good candidate. Lastly, a_4 has work experience and is a fast learner, but will be troublesome within the group, a Moderate candidate. Clearly, the labelling is understandable and intuitive, but not very rigourous. These evaluations are summarised in Figure 1.2(a).

(a) Evaluation of candidates.

	c_1	c_2	c_3	rank
a_1	$-$	$-$	$+$	B(ad)
a_2	$+$	$-$	$-$	M(oderate)
a_3	$-$	$+$	$+$	G(ood)
a_4	$+$	$+$	$-$	M(oderate)

(b) Classification tree.

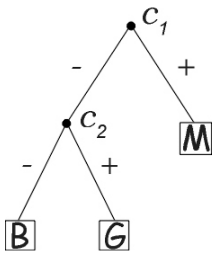


Figure 1.2: Candidate evaluations.

Remark this is just the same table as in Figure 1.1(a), which means we have again the same classification tree, as shown in Figure 1.2(b). It turns out that the best possible candidate, namely someone with a lot of working experience, who is a fast learner and fits well in the group, in other words, a candidate with evaluations $(+, +, +)$, is evaluated as Moderate by this tree. However, another person who is only capable of learning fast, but has no experience and will not get along with his/her colleagues, thus having evaluations $(-, +, -)$, ends up in the class labelled Good. This is in contradiction with every fiber of intuition we possess about such problems: we expect that the best candidate gets a higher rank than this other candidate, because the former one is obviously to be preferred over the latter one.
In general, we expect that a higher score on one or more of the criteria results in a better or equal ranking. This is just stating that we expect a *monotone* increase of the final evaluation w.r.t. to the partial evaluations. And this is the point where classification trees (and other classification algorithms) fail: they cannot guarantee that this property of monotonicity holds.

1.2.3 Problems of an ordinal nature

or·di·nal. *Adjective:* Being of a specified position in a numbered series. – *The American Heritage® Dictionary of the English Language, Fourth Edition.*

Basic scales. The basic types of scales from the representational theory of measurement are the same ones we encounter in the techniques for dealing with supervised learning. They are the nominal, ordinal, interval and ratio scales. In a *nominal scale*, the different values have no real relation between them, like {Yellow, Blue, Red}. In an *ordinal scale*, only the order between its values are known, for example {Small, Average, Big}. In an *interval scale*, it is possible to compare differences between values, i.e. there exists a unit to the scale. Finally, in a *ratio scale*, two values can be compared in an absolute manner, in other words, it is like an interval scale but with some "true"[1] zero. Frequently, the term *numerical scale* is employed to denominate both interval and ratio scale.

Discreteness versus continuity. In this thesis, we will mainly bother about ordinal scales because this is the type of scale we predominantly encounter in ranking problems. This poses again some problems, because most of the previous work in supervised learning is based on nominal scales or numerical ones. Although converting an ordinal scale into a nominal one is possible in a mathematically sound way, it results in a too important loss of information. On the other hand, we do not have enough information to justify a sound transformation from an ordinal (discrete) scale into a numerical (continuous) one. That is why a popular assumption (typically coming from utility theory[2], e.g. [57, 66] concerning ordinal classification) is that an ordinal variable is the result of a coarsely measured latent continuous variable, each value on the ordinal scale delineating an interval on the continuous scale (see Section 3.5.1, p. 75 for more details). There is nothing wrong with this until you start to add, subtract, multiply and divide these transformed figures, for example by calculating their mean value. Consider the following simple example: assume we are working with the ordinal scale {Bad, Ok, Perfect}. It seems acceptable to allow a transformation into a numerical scale such that the mean of Bad and Perfect results in Ok. However, if we consider the extended scale {Bad, Ok, Better Than Ok, Good, Perfect}, you might reconsider this and even be of the opinion that it is not possible to take the mean of Bad and Perfect because these values are too extreme, that the idea of the mean of these values cannot be filled in meaningfully.

[1] "*The distinctive feature of a ratio scale is that it has an origin defined by a dominating theory* [110, p. 25]".

[2] See http://cepa.newschool.edu/het/essays/uncert/choiceref.htm for a selection of references.

1.3 Problem solving

We already know *what* we want to achieve: creating learning algorithms for ranking, the question that remains pending is *how* we will accomplish this. We could start from scratch and endeavour the construction of a ranking algorithm without regard of history and existing knowledge. This would undoubtedly be a very instructive approach, but the pitfall of reinventing the wheel and loosing precious time is always lurking around during such attempts. Moreover, it is very likely that the reinvented wheel will not roll as smoothly as the one chiselled by mankind during past decades if not centuries. Another approach would be to start from existing classification algorithms, and mend, enhance or adapt them so they can handle ranking problems. This demeanour is the most common one, but now, the risk is to end up with some tinkered construction. You can always start with a wheel to create a sphere, but wouldn't it be better to shape a sphere from rudimentary material by using the tools and knowledge gained during years of wheel-making?

1.3.1 Fundamental approach

Understanding a problem is often the first step toward solving it.

This is the creed every layer in this thesis complies with. Of course, you can indulge in this quest for knowledge and understanding as far as you want. For example, we already figured out that the main problem for supervised ranking lies in respecting the monotonicity inherent to the ranking, so we could content ourselves with somehow enforcing monotonicity upon the output of the classifier. However, we succumbed to the call of the deeper fundamentals which are concealed well inside the problem behind an elusive veil of symbols and mathematics. Evidently, a thesis cannot pretend to be more than but one pebble in a larger heap. Therefore, we do not divulge all philosophically tinted treatises that could be related to our topic of interest, but merely skimmer across their surfaces and highlight the thoughts that can be of help in our research.

Overseeing the whole. If we probe deeper into the being of induction methods such as classification trees and the like, we find that they are based on a mathematical model. The original harbour of the problem statement can be traced back even further, to the real world that is represented by this model, see also Figure 1.3. Thus, before meddling with the final stage, it would be more than interesting to investigate the real world problem statement and its representation in a mathematical model.

Understanding the parts. To seize the whole, it is usually clarifying to understand the parts from which it is built. Figure 1.3 embraces three main pillars: the real world (e.g. what is meant by “classification”, what is meant be “learning”?), its representation (how is this idea of classification and learning translated into a mathematical framework?), and lastly the actual methods and algorithms.

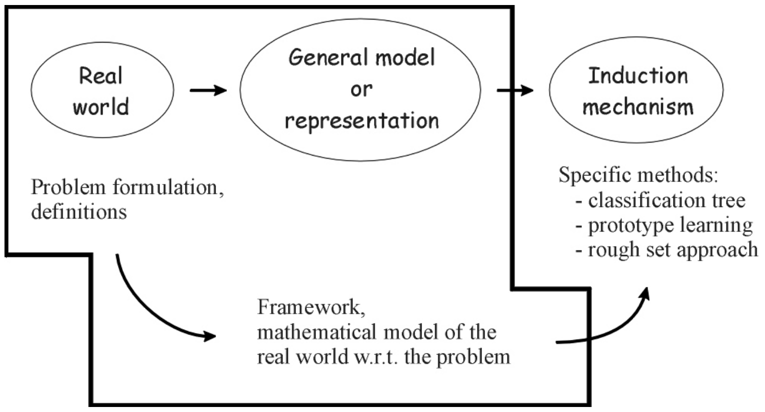

Real world
General model or representation
Induction mechanism
Problem formulation, definitions
Specific methods:
- classification tree
- prototype learning
- rough set approach
Framework, mathematical model of the real world w.r.t. the problem

Figure 1.3: Overseeing the whole.

Comprehending the interactions. Because the whole is more than the sum of its parts, we also need to have a clear view on the interactions between these parts. We already saw that rankings are a kind of variation upon classifications, but what are the effects of this on the representation, and how do these effects propagate to the final learning methods? By our comprehension of the interactions between the different parts, the consequences of altering a definition in one part become more manageable in the other parts.

Grasping the reasons behind. Far more interesting and instructive than the *how* is the *why* of matters. Typically, in a mathematically oriented thesis, you expect to find theorems and proofs. However, that is the "easy" part. It can be as horribly simple to prove something as it can be horribly difficult, but what is ever tougher and harder, is to understand why it can be proven. For example, it is relatively simple to demonstrate that the earth is spherical, and it was done already in ancient times (e.g. by Eratosthenes in 250 B.C. [70]). However, explaining why the earth has a spherical shape is a whole other matter. You can not just understand why the earth has a spherical form by only considering the earth as a singled out object standing all by itself. Rather, you need to situate the earth in the universe, floating along with other stellar bodies. You need to investigate the forces of nature, like gravity, understand the concepts of mass and energy, until finally, it becomes self-evident why the earth had to be spherical, why it could not have been otherwise.

Act non-invasive. In everything we do, we try to follow the non-invasive attitude advocated in [45]. Taking on a very gingerly demeanour, we try not to make any assumptions that can not be backed, or perform operations that do not come rolling out of our understanding of the problem. Even if it may seem clear that something needs to be done, we shun from doing it unless we can explain why it needs to be done and show what are the consequences of actually doing it[3]. This careful attitude stems from the knowledge that even the sharpest intuition may prove wrong in the end, and conducting a massive amount of experiments may still give a false impression (though intuition and experiments form certainly important facets of research).

1.3.2 Perception

Perception is the backbone of all our thoughts and actions.

Now we enter the realm of philosophical reflection and meditation. Our perception of reality is usually cloaked in deceptive simplicity, and it is worth while to take a moment to linger and ponder on it.

Epistemology. A classification should be viewed inside some epistemological theory[4]. Basically there are four methods of classification based on the following epistemological theories [60]: 1) *empiricism*: all knowledge comes from the senses; 2) *rationalism*: all knowledge comes from thinking; 3) *historicism*: all knowledge depends on biological, cultural, social and individual developed conditions; 4) *pragmatism*: a variant of historicism claiming that the analyses of goals and values must play an important role in the establishing of knowledge.
Supervised learning has historically been interested in the first two of these theories: empiricism (like neural nets) and rationalism (symbolic methods). Multi-criteria decision aid, is more keen on historicism (preference modelling) and pragmatism (multi attribute utility theory).

Syntax, semantics and praxis. The study of human communication can be divided into the three terrains of *syntax*, *semantics* and *praxis* [121]. The first terrain, the *syntax*, is concerned with the problems of information transmission, i.e. encoding, noise, redundancy, etc . . . The meaning of the information symbols is of no importance. The *semantics* is primarily about meanings. For messages (syntactically constructed sequences of symbols) to make sense, it is necessary to have some convention about their meaning before the communication takes place. In other words, besides semantics, there is no other correlation between a word and what it stands for. Finally, communication influences behaviour.
Now consider the following phrase from the description of supervised learning: *"The training data consists of pairs of input objects (typically vectors), and desired outputs."* This one sentence harbours quite some hidden layers. Firstly, the word *"object"* obviously refers to some kind of entity or idea in the real world. However, when put in some kind of data record, the "objects" in question are pinpointed by means of so-called *digital* communication [121], i.e. by *naming* them. Names or

[3]Based on this point of view, Chapter 7 is called "The basics of ranking trees", and not simply "Ranking trees" because the final algorithm mentioned in that chapter is just a melting pot of the most basic ideas and understandings concerning decision trees for the supervised ranking problem.

[4]Epistemology is a branch in philosophy concerning the theory of knowledge

words are just arbitrary signs that are handled according to the logical *syntax* of the language. The fact that a word refers to some real world[5] object is merely a matter of a *semantical* convention of the language. Secondly, the remark *"(typically vectors)"* refers to some secondary representation of the objects, by means of certain properties they possess. Again, syntax and semantics surge to the front, and to make matters even more complicated, they go hand in hand with the representational theory of measurement. Indeed, there needs to be agreement about what is meant by the describing properties (q_i), and what is meant by the relation between the symbols (x_i, y_i) used to describe the objects (see Table 1.1). On top of that,

	property q_1	property q_2
object o_1	x_1	x_2
object o_2	y_1	y_2

Table 1.1: Syntax and semantics in the training data.

this small two word remark supposes that a scale is attached to these describing symbols, in other words, that they are in fact measurements of some kind. Here we see a first and obvious example of how different semantics are poured into different mathematical models[6]. Also remark that the same is true for the *"desired output"*. Clearly, the importance of semantics can hardly be overestimated. And because the concept of ranking is rooted in multi-criteria decision aid which flirts with a mathematics that is loaded with semantical ideas, we have to be twice as careful in order to avoid nonsensical results and/or the mixing of incompatible semantical ideas. Moreover, starting from a sound semantical foundation will also enable us to understand more easily the "why" of matters.
One final word of caution is at place: the semantics of a notion should not be confounded with the properties of this notion. A property can somehow be verified, but semantics are purely a convention about meaning.

Objective and subjective reasoning. Closely related to the previous topics is the discussion about objective and subjective information. Empiricism and rationalism are schools embracing the principle of objectivity. On the other hand, historicism and pragmatism stress the importance of subjective information, which may well be in conflict with logical reasoning. Since we want to reconcile these approaches, we need to make some choices. Our approach will be to incorporate ideas from multi-criteria decision aid into machine learning. In doing so, we basically try to reshape subjective information that leads apparently to logical inconsistencies

[5]We assume there exists something as a "real world", although this is again a matter of philosophical debate [73].

[6]In many cases, the semantics are based on the measurement scale used, but it may well happen that the measurement scale is only an approximation of the intended semantics. For example measuring size on a ratio scale, versus measuring quality (e.g. Poor, Acceptable, Decent, Good) on an ordinal scale. In the first case, the proportion between the sizes is explained by the ratio scale, however, in the second case, it is very likely that the ordinal scale is in fact a bit coarser than what is actually intended by the enumeration Poor, Acceptable, Decent, Good.

into an acceptable objective form. Moreover, in compliance with our non-invasive approach, we choose to do so with the utmost respect for the given data.

1.4 How to read this thesis

All chapters are self-contained and can therefore be read independently from each other. To achieve this goal, all necessary concepts are repeated in each chapter before the sections they are used in. This is either done in a paragraph named "**notions and conventions**", or it is indicated in the margin. Moreover, all notions and symbols are repeated in the margin, **like this**, when they first appear in each chapter. This means you never need to thumb back very far in case you have forgotten the meaning of one. It also helps you in skipping passages: if the notions in the margin seem familiar, just jump to the next paragraph or subsection.

LIKE THIS.

Even if all chapters can be read (more or less) individually, they do make up a continued story. The outline of this thesis is depicted in Figure 1.4.

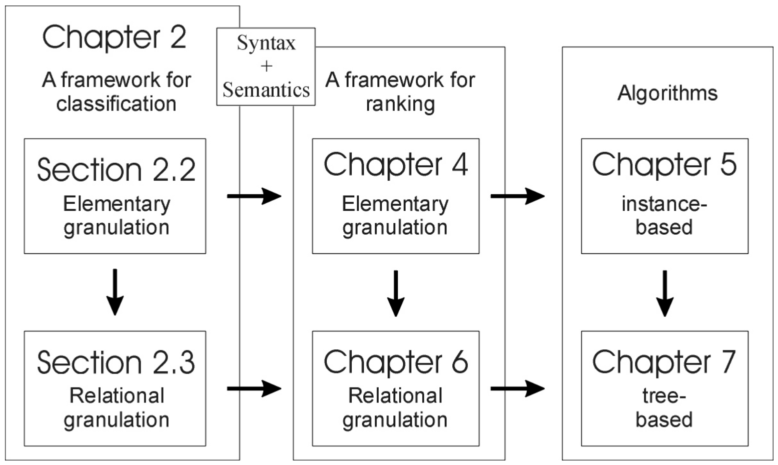

Chapter 2
A framework for classification
Syntax + Semantics
A framework for ranking
Algorithms
Section 2.2 Elementary granulation
Chapter 4 Elementary granulation
Chapter 5 instance-based
Section 2.3 Relational granulation
Chapter 6 Relational granulation
Chapter 7 tree-based

Figure 1.4: Outline of this thesis.

Of course, every thesis worthy of that name has some kind of a literature study, and this one will not be the exception to this common rule. Chapter 3 fills in this gap in the outline given above and tempts a summary review of existing approaches for supervised ranking.

APPENDIX

Learning algorithms for classification

Because we want this thesis to be self-contained, we add this appendix discussing more at length decision tree algorithms and the rough set methodology. People already familiar with these can therefore easily skip this chapter, although they might want to read Appendix 1.C, since it consists of a more personal point of view concerning how far we feel is the reach of rough sets and where do we draw the imaginary line between methods based on rough sets and methods based on the same philosophy underlying rough sets.

1.A Decision trees

Introduction. Creating a decision tree is usually done in two steps: first an overly large tree is grown (Appendix 1.A.3), and afterwards, this tree is pruned to a smaller size (Appendix 1.A.4). For a multi-disciplinary survey, see [79, 80].

1.A.1 History

Decision trees have had a quite long conception time starting in the late fifties by the work of Hoveland and Hunt under the name CLS (*Concept Learning Systems*) and in the early sixties with the AID (*Automatic Interaction Detection*) program of Morgan and Sonquist for regression, and its successor THAID of Morgan and Messenger in the early seventies for classification (TH refers to theta, a letter that was used to indicate the proportion of correctly predicted cases for the entire (training or test) set). Decision trees were then rediscovered and studied by several people in diverse disciplines. The standard algorithms resulting from these efforts are CART (*Classification and Regression Trees*, [23]) coming from statistics and ID3 (*Induction of Decision Trees*, [92]) which originated from the discipline of machine learning, together with its successor C4.5 [93].
From that moment on, a multitude of adaptations and new decision tree algorithms were suggested, including faster algorithms [71], scalable algorithms [62, 82], and algorithms capable of dealing with relational information [63, 64, 67, 69].

1.A.2 The decision algorithm

The easiness to understand a decision tree is one of its most appealing trumps. Consider for example the “student”-tree in Figure 1.5. The meaning of it is self-explaining: if an object needs to be classified, just answer the question at the top of the tree, and follow the branch that contains the right answer. Continue like this with the next questions and answers, until finally, it is stated to which class the object belongs. So, a decision tree is just a representation of a set of rules that form a hierarchical partition of the data.

Notions and conventions. A **tree** T consists of a series of **node**s and **branch**es. End-nodes are called **leaves**, the other ones are called **inner node**s. The branches connect the nodes. Inner nodes are labelled with a **split** question. The branches are labelled with the possible answers to these questions. A leaf t is labelled with the response that is associated to all the objects falling into t. This response can be a class label and/or a distribution over the different class labels, or, in the case a continuous function is learned, it can be a real number. The set of leaves of a tree T is denoted by $\widetilde{T}$. The subtree of T starting at the node t is denoted by T_t. See also Figure 1.5, where also the notions of *root*, *parent* and *children* are explained.

TREE.
NODE.
BRANCH.
LEAF.
INNER NODE.
SPLIT.

$\widetilde{T}$
T_t

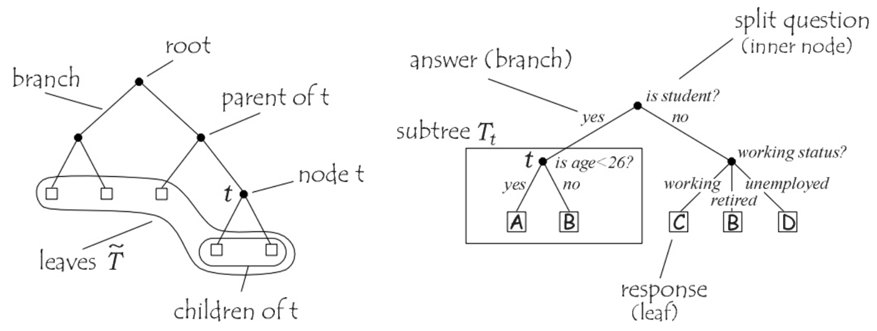


Figure 1.5: Terminology of decision trees.

Because a node in a classification tree gathers a set of objects, it is possible to define a probability distribution over the class labels for each node. For a leaf t, we denote $p(t)$ the estimated probability that an object falls into the leaf t, and, if we denote the set of classes by $\mathcal{L} = \{1, \ldots, k\}$, then we denote by $p(i|t)$ the estimated probability that an object falling into t belongs to class i.

$p(t)$
$p(i|t)$

1.A.3 Model selection: growing the tree

Growing a tree consists of the recursive splitting of the tree, i.e. finding appropriate split questions from a set of possible split questions. Usually, a greedy heuristic is followed: the best split for each node is taken without taking into consideration possible splits of the resulting children. Obviously, a greedy heuristic cannot guarantee to find the optimal tree, but it has been experimentally ascertained that it delivers near-optimal results [79].

There exist several measures to gauge the power of a split question on the given data, with a background ranging from information theory to statistics. The most widely used measures for classification trees are the so-called impurity measures.

IMPURITY MEASURE.

Impurity measures. An **impurity measure** is a non-negative function ϕ_n from the set of probability distributions over the class labels $\mathcal{L} = \{1, \ldots, k\}$ to $\mathbb{R}$ such that

- ϕ_n reaches its maximum in $(\frac{1}{n}, \ldots, \frac{1}{n})$, i.e. the objects are distributed as heterogeneous as possible over the classes;
- ϕ_n is zero if there is no uncertainty, i.e. all objects belong to the same class:
$$\phi_n(1, 0, \ldots, 0) = \phi_n(0, 1, 0, \ldots, 0) = \phi_n(0, \ldots, 0, 1) = 0\,;$$
- ϕ_n is symmetric: for all permutations σ of $\{1, \ldots, n\}$, we have
$$\phi_n(p_1, \ldots, p_n) = \phi_n(p_{\sigma(1)}, \ldots, p_{\sigma(n)})\,.$$

Applied to a node of a tree, this becomes

$$\phi(t) := \phi_n(p(1|t), \ldots, p(n|t))\,.$$

The impurity of the whole tree T is then defined as the weighted sum of all leaf-impurities:

$$\phi(T) = \sum_{t \in \widetilde{T}} p(t)\phi(t)\,.$$

In order to obtain a higher discrimination power with dropping uncertainty, sometimes the condition of strict concavity is added:

$$\phi_n''(p_1, \ldots, p_n) < 0\,.$$

SHANNON ENTROPY H. GINI DIVERSITY INDEX G.

The most frequently used impurity measures, the **Shannon entropy** and the **Gini diversity index**

$$H(t) = -\sum_i p(i|t) \log_2 p(i|t) \quad \text{and} \quad G(t) = \sum_{i \neq j} p(i|t)p(j|t)\,,$$

oblige this last condition. Their graph is shown in Figure 1.6 for the case of two classes only.

1.A.4 Model selection: pruning the tree

Overfitting. The idea of pruning (the opposite of splitting) is to avoid the problem of overfitting the problem. When the tree gets larger and larger, the leaves contain less and less learning objects. As a consequence, the global impurity of the tree may drop, but the number of representant objects inside the leaves becomes too small to lead to a well-founded conclusion about the class distribution inside the leaves. So, while the error on the learning sample may be very low, the error on an independent test sample may be quite high. The pruning strategy is to first grow an overly large tree, and then to remove certain splits, hence producing smaller trees with leaves taking up larger portions of the data. Remark however that Shaffer [98] made clear that there are no *"statistical reasons for believing that these overfitting avoidance strategies do increase accuracy"*. Nevertheless, smaller trees are at least easier to interpret.

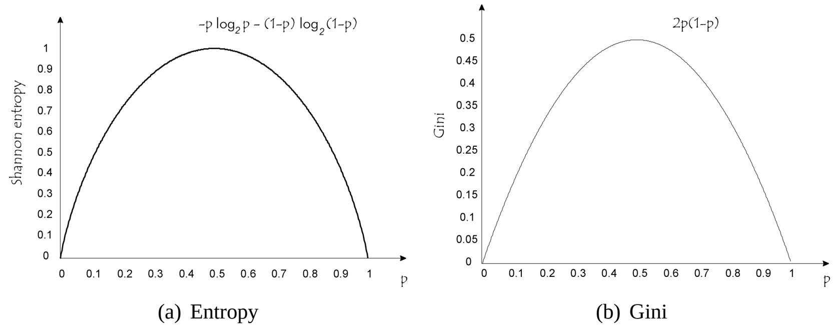


Figure 1.6: Shannon entropy and Gini diversity index, 2-class problem.

Pruning methods. There are two types of pruning strategies: pre- and post pruning. Pre-pruning curtails the tender growing of a tree, while post-pruning first grows a large tree $T_{\max}$, which is afterwards pruned carefully into shape. There are two types of post-pruning strategies, (i) single-step: run through the nodes (either bottom-up or top-down), and decide at each node whether to prune it, or (ii) two-step: first generate a set of pruned trees, and then select one of them. For an overview of these methods, see for example [24, 113].

1.B Rough set methodology

1.B.1 History

The introduction of rough sets in 1982 [85] can be attributed to one person, namely Pawlak. However, the date 1982 only refers to the moment the name "rough set" was jotted down. The very beginning can be traced back to the late sixties (Salton's *Automatic Information Organisation and Retrieval*), and the early seventies where Pawlak (later together with Marek) published a mathematical model for attribute based information systems. The work done by the Information System Group in Warsaw on this model, finally led to the paper [84] where most of the ideas underlying rough sets are exhibited (but without the terminology proper to the rough set methodology).
Note that decision trees were originally conceived to deal with prediction, but later on, also also applications in information retrieval (e.g. [6]) have been found. On the other hand, RSDA started as a method for information retrieval [84], and has only afterwards been extended to deal with prediction.

1.B.2 Rough sets

EQUIVALENCE RELATION.

Notions and conventions. An **equivalence relation** is a reflexive, symmetric and transitive relation.

INFORMATION SYSTEM $\mathbf{I} = \langle \Omega, Q, \mathcal{X}_Q, \varphi \rangle$.

Approximation space. The input data are represented by an **information system** [84] $\mathbf{I} = \langle \Omega, Q, \mathcal{X}_Q, \varphi \rangle$, where Ω is a finite set of objects, $Q = \{q_1, \dots, q_n\}$ a finite set of attribute names, where each $q \in Q$ has an associated set of values $\mathcal{X}_q$, $\mathcal{X}_Q = \bigcup_{q \in Q} \mathcal{X}_q$, and $\varphi : \Omega \times Q \to \mathcal{X}_Q$ such that $\varphi(a, q) \in \mathcal{X}_q$.
What is interesting to these notations, is that they make explicit the human interference when dealing with the real world: the objects as well as their representation are kept clearly visible. This helps in avoiding making mistakes such as identifying an object a with its representation $\mathbf{a} = (\varphi(a, q_1), \dots, \varphi(a, q_n)) \in \mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_{q_i}$. To alleviate notations, we revert to the less cumbersome notations $\mathbf{I} = \langle \Omega, Q \rangle$, and $q(a) := \varphi(q, a)$. The basic observation underlying RSDA is that objects may be **indiscernible** due to the limited availability of information. This leads to the introduction of the **indiscernibility relation** $\mathcal{I} \subseteq \Omega \times \Omega$ as

INDISCERNIBLE.

INDISCERNIBILITY RELATION $\mathcal{I}$.

$$a\mathcal{I}b \iff (\forall q \in Q)(q(a) = q(b)) .$$

If $a\mathcal{I}b$, then the objects a and b are indiscernible from each other w.r.t. the attributes from Q. We know by their names (a, b) that they are not one and the same object, but we cannot tell them apart by merely considering their feature vectors $(\mathbf{a}, \mathbf{b})$ in $\mathcal{X}$.
A pair $\langle \Omega, \Pi \rangle$, with Ω a finite set and Π an equivalence relation, is called an **approximation space**[7]. It is as if peering at the set Ω through a filter, Π, blurring out some of the details. When looking at $a \in \Omega$, we can only see it belongs to the equivalence class $\Pi(a) = \{b \in \Omega \mid a\Pi b\}$, without being able to actually single out the object a itself. It is clear that the indiscernibility relation $\mathcal{I}$ is an equivalence relation and therefore defines an approximation space. The induced equivalence classes are called **(information) granules** in this context. Remark that each granule is determined by exactly one feature vector in the measurement space $\mathcal{X}$ (although not every vector of $\mathcal{X}$ needs to corresponds to a granule because the image of Ω under λ may be only a subset of $\mathcal{X}$).

APPROXIMATION SPACE.

(INFORMATION) GRANULES.

For each subset of attributes $P \subseteq Q$, a new information system $\mathbf{I}_P = \langle \Omega, P \rangle$ can be devised with associated indiscernibility relation $\mathcal{I}_P$:

$\mathbf{I}_P, \mathcal{I}_P$

$$a\mathcal{I}_P b \iff (\forall q \in P)(q(a) = q(b)) .$$

So, each $P \subseteq Q$ also defines an approximation space $\langle \Omega, \mathcal{I}_P \rangle$. The partitioning of the measurement space $\mathcal{X}$ induced from such an indiscernibility relation is nothing else but a grid partitioning and corresponds to $\prod_{q \in P} \mathcal{X}_q$. Therefore, with some abuse of language, we will call an approximation space of the type $\langle \Omega, \mathcal{I}_P \rangle$ simply a **grid**.

GRID.

[7]Sometimes this is called the Pawlak approach to rough set theory, a treatise of how it links with *abstract approximation spaces* can be found in [28].

"Standard" rough sets. The granularity of information accounts for the fact that class boundaries (remember that classes are essentially subsets of the object space) in the training sample $\mathcal{S}$ cannot be determined sharply. Instead, through the blurry spectacles of the indiscernibility relation, we can only pinpoint the so-called *lower* and *upper approximations* of the classes, containing respectively the objects from $\mathcal{S}$ that we can distinguish as certainly belonging to a class, and the objects that only possibly belong to it, i.e. not containing the objects that certainly do not belong to the class. More formally, let $\langle \mathcal{S}, \Pi \rangle$ be an approximation space with associated set of granules $\mathcal{X}_\Pi$, and $A \subseteq \mathcal{S}$, then

$$\underline{A}_\Pi = \bigcup \{a \in \mathcal{S} \mid \Pi(a) \subseteq A\}$$

is called the **lower approximation** of A, and

LOWER APPROXIMATION $\underline{A}_\Pi$.

$$\overline{A}^\Pi = \bigcup \{a \in \mathcal{S} \mid \Pi(a) \cap A \neq \emptyset\}$$

the **upper approximation** of A. See also Figure 1.7.

UPPER APPROXIMATION $\overline{A}^\Pi$.

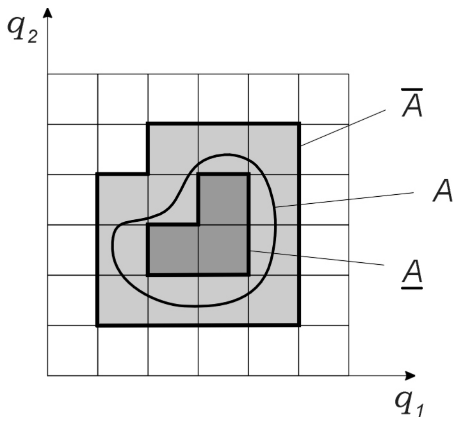


Figure 1.7: lower and upper approximation.

If Π is understood, we usually omit the subscript (superscript). In general, Π corresponds to $\mathcal{I}_P$ for some subset of attributes $P \subseteq Q$. In that case, we also write $\underline{A}_P$ and $\overline{A}^P$. A **rough set** (w.r.t. the approximation space $\langle \mathcal{S}, \Pi \rangle$) is a pair $\left\langle \underline{A}, \overline{A} \right\rangle$. The two following properties are essential:

ROUGH SET.

(i) rough set property: $\underline{A} \subseteq A \subseteq \overline{A}$,

(ii) complementarity property: $\mathcal{S} \setminus \overline{A} = \underline{\mathcal{S} \setminus A}$ and $\mathcal{S} \setminus \underline{A} = \overline{\mathcal{S} \setminus A}$,

where (i) explains the names *lower* and *upper approximation* and *rough set*.

The variable precision rough sets model. In [126], a different and more flexible scheme for defining upper and lower approximations was proposed. The core idea was to extend the inclusion relation present in the lower approximation.
It is based on the following inclusion function:

$$c(X,Y) = \begin{cases} \frac{|X \cap Y|}{|X|} & \text{, if } |X| > 0 \\ 1 & \text{, if } |X| = 0\,. \end{cases}$$

For some $0 \leq \beta < 0.5$, the lower and upper approximation are now defined as

$$\underline{A}_\Pi = \{a \in \mathcal{S} \mid c(\Pi(a), A) \geq 1 - \beta\} \quad \text{and} \quad \overline{A}^\Pi = \{a \in \mathcal{S} \mid c(\Pi(a), A) \geq \beta\}\,.$$

If $\beta = 0$, these expressions result back into the "standard" definitions of lower and upper approximation.

1.B.3 The decision algorithm.

For easier reading, we will identify the granules $\Pi(a) \subseteq \mathcal{S}$ with their counterparts in $\mathcal{X}$, and we do the same for the lower and upper approximations that are built from these granules.
There are two approaches:

(i) The decisions are made directly based on the blocks $\mathcal{X}_\Pi(a) \in \mathcal{X}_\Pi$ induced by the granules $\Pi(a)$ defining the lower an upper approximations, e.g. [36, 45]:

  (a) $\mathbf{b} = (q_1(b), \ldots, q_n(b)) \in \Pi(a) \subseteq \underline{A}$ for some $a \in \mathcal{S}$ implies that b *certainly* belongs to A,

  (b) $\mathbf{b} \in \Pi(a) \subseteq \overline{A}$ for some $a \in \mathcal{S}$ implies that b *possibly* belongs to A.

  (c) $\mathbf{b} \in \overline{\mathcal{S} \setminus A}$ implies that b does *certainly not* belong to A,

  Remark that, if Π corresponds to $\mathcal{I}_P$ for subset of $P = \{q_1, \ldots, q_k\} \subseteq Q$, then "*if* $\mathbf{b} \in \Pi(a)$" can be rewritten as "*if* $q_1(b) = v_{q_1}$ *and* … *and* $q_n(b) = v_{q_k}$", with $v_q \in \mathcal{X}_q$.

(ii) A set of rules is generated from the lower and upper approximations, leading to certain and possible rules e.g. [29, 109]. However, in contrast with the above approach, certain rules may overlap each other.

To conclude, remark that $\bigcup_{i \in \mathcal{L}} \underline{C_i}$ may not cover the universe $\mathcal{S}$, in which case non-determinism occurs, while on the other hand we always have $\bigcup_{i \in \mathcal{L}} \overline{C_i} = \mathcal{S}$.

GENERALISED DECISION.

The generalised decision. Assume we are given a decision system, i.e. an information system $\langle \mathcal{S}, Q \rangle$ and a decision function $d : \mathcal{S} \to \mathcal{L}$. It is possible to replace d by a new function $\delta_Q : \mathcal{S} \to 2^\mathcal{L}$, called the **generalised decision**, such that the resulting decision system becomes deterministic, meaning $|\delta_Q(x)| = 1$ for all $x \in \mathcal{S}$, by defining $\delta_Q(x) = \{i \in \mathcal{L} \mid (\exists x' \in \mathcal{S})(x \mathcal{I}_Q x' \wedge d(x') = i\}$. The objects from $\mathcal{S}$ can now be partitioned into subsets, each containing objects described by the same value of the generalised decision function.

1.B.4 Model selection: the standard rough set approach

The core idea is to find an approximation space, more particular a grid $\langle \mathcal{S}, \mathcal{I}_P \rangle$, using the least possible attributes while still maintaining the same *approximation quality* as with the approximation space $\langle \mathcal{S}, \mathcal{I} \rangle$. This is done in a bottom-up (i.e. specific to general) search through the space of possible grids. It has to be mentioned that a greedy bottom-up search did not lead to satisfactory results, and therefore other heuristics have been investigated.

The most common definition of a *reduct* is the one based on a measure called the **quality of approximation** defined as

QUALITY OF APPROXIMATION γ_P.

$$\gamma_P(\mathcal{S}, d) := \sum_{i \in \mathcal{L}} \frac{|C_{i_P}|}{|\mathcal{S}|} ,$$

i.e. the proportion of the number of correctly reclassified sample objects and the total number of samples. A **reduct** is any (non-empty) subset of attributes $P \subseteq Q$ that such that $\gamma_P(\mathcal{S}, d) = \gamma_Q(\mathcal{S}, d)$. A **minimal reduct** has the additional property that $\gamma_{P'}(\mathcal{S}, d) < \gamma_Q(\mathcal{S}, d)$ for any (non-empty) $P' \subseteq P$. Once a reduct has been chosen, it is used to define the partition Π on $\mathcal{S}$ used in the decision algorithm.

REDUCT.

MINIMAL REDUCT.

It is known that such reducts are not very good at predicting the class of new unseen objects (overfitting). Therefore, alternatives are proposed, such as *dynamic reducts* [7].

A simple example. Consider the data shown in Figure 1.8.

(a) Table.

	q_1	q_2	d
a_1	1	1	A
a_2	1	1	A
a_3	1	2	A
a_4	2	1	A
a_5	2	1	B

	q_1	q_2	d
a_6	2	1	A
a_7	2	2	B
a_8	2	2	A
a_9	3	1	B
a_{10}	3	2	B

(b) Approximation space.

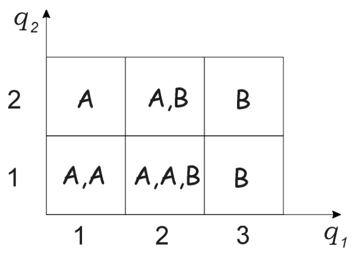


Figure 1.8: A simple example.

We have that $\gamma_{\{q_1,q_2\}}(\mathcal{S}, d) = \frac{2}{10} + \frac{1}{10} + \frac{1}{10} + \frac{1}{10} = \frac{1}{2}$, $\gamma_{\{q_1\}}(\mathcal{S}, d) = \frac{3}{10} + \frac{2}{10} = \frac{1}{2}$, and $\gamma_{\{q_2\}}(\mathcal{S}, d) = 0$. So, we find that $\{q_1\}$ is a reduct, and that it is a minimal one.

1.C A note on rough set methodology

1.C.1 What is the rough set approach?

What is the *rough (set) approach*? This seems a difficult question to me. If you ask “What are decision trees, what are neural nets, what are nearest neighbour methods?”, the answers are clear and straightforward. The same is true if one asks “What is fuzzy set theory, what is nonstandard analysis,. . . ?”.
In general, it seems that a publication is catalogued as dealing with rough sets if it elaborates on either the subject of granularity, lower and upper approximations, or, in the context of data mining, searching the space of possible grids[8]. Of course, an enumeration of all subjects of “rough set articles” does not provide a suitable reply to the question posed above.
Maybe it could be put as follows: the rough set approach is any approach based on rough set theory. This answer naturally brings forth a second question: “What is rough set theory?”

Granularity and approximations. The notion of an approximation space is one of the fundamental concepts in rough set theory. It is usually defined as a couple (Ω, R), where R is just a relation (mostly the *indiscernibility relation* (see p. 18)) on the set Ω. The sets $R(a)$ are called granules and we denote $\Omega_R = \{R(a) \mid a \in \Omega\}$. However, we feel that a distinction should be made between granular spaces and approximation spaces. A **granular space** should be defined as the previous mentioned couple (Ω, R), while an **approximation space** is a granular space together with two mappings on the powerset of Ω, an **inner approximation mapping** $i : 2^\Omega \to \Omega_R$ and an **outer approximation mapping** $o : 2^\Omega \to \Omega_R$ such that $i(A) \subseteq A \subseteq o(A)$ for any $A \subseteq \Omega$. For a more formal and concise definition of (and elaboration on) the notion of abstract approximation spaces, see [28] . Rough set theory is then the theory of abstract approximation spaces and its concrete realisations.

GRANULAR SPACE. APPROXIMATION SPACE. INNER APPROXIMATION MAPPING. OUTER APPROXIMATION MAPPING.

In short, working with granules is not enough to characterise a method as following the rough set approach, not even if it deals with the “classificatory analysis of data tables [65]”, since clearly decision trees fit such a description. To earn the label rough set methodology, some notion of inner (=lower) and/or outer (=upper) approximation must be involved, either directly or within some statistic used, such as the *quality of approximation* (see p. 21). However, the rough set community should be accredited for stressing the importance of granulation which lies at the core of their work.

Grids. A grid can be viewed as a granular space. The merit of meticulous investigation of the search space of grids [95] in classification problems is certainly to

[8]This enumeration is not exhaustive, but suffices for our present needs.

be attributed to the development of rough set theory. Still, if there is no reference to the concepts of inner/outer approximations, we see no reason to say it is part of the rough set approach. Rather, it is possible to use a *rough version* of the method, for example by defining the notion of reduct using the quality of approximation.
The search of the space of possible grids lies at the same level as the search of the space of possible decision trees. The former is a bottom-up approach, while the latter is a top-down version of the same idea. Both can be turned into rules and further investigated using the same techniques. Both can be based on measures stemming from rough set theory [65, 82], but also from information theory [93, 106].

1.C.2 Conclusion

We feel that the name *rough set approach* is used a bit too easily. Comparable to some extent to the way that some people call something fuzzy as soon as the real interval $[0, 1]$ is used, while in fact one also needs that one of the three semantics of fuzzy sets (similarity, incompleteness (or vagueness), preference) is applicable. Likewise, rough sets is not just granularity, it needs a specific use of this granularity: identifying inner/outer approximations. Still, the rough set methodology was the first to stress the importance of granularity.

Interlude

TRANSLATIONS

Let me issue a small warning: "Do not be too diligent!"

THE FRENCH CONFERENCE. *What a marvellous idea it seemed at the time I saw the announcement for some French conference. Yes, I thought to myself, why not practice my French writing a bit and submit an article in the native language of the conference? So, this chapter was originally written in French as a submission to this conference. I typed the paper and got some very nice assistance of Céline, who took the effort of correcting my sometimes awkward French constructions*[9]*.*

THE ENGLISH THESIS. *What a silly idea to write a text in French! How could I have been so foolish? Now I have to translate the whole darn thing back into English. And I did not even get accepted to the conference!*

The moral to this story: "Do not be too diligent!"

[9]Céline, thank you again :o)

Chapter 2

A framework for classification

2.1 Introduction

As we mentioned in Section 1.3.1, it is essential to understand the basics to their full extent before we should even consider building on their foundations, further embellishing and ornamenting them. That is why we will probe deeper into the fundamentals of classification (within the context of supervised learning). Nothing shockingly new will come about, in the end, all we do is analyse the whole, break it down in different parts, name them, and put them together again, restating what is (or at least seems) obvious, while extending some other parts. Still, this approach will enable us to put decision trees and rough set analysis in a slightly different perspective which will be useful in the second part of this thesis.

The first section rephrases the essential setup needed to mathematically grasp (represent) the idea of classification and supervised learning. The core idea can be traced back to what is called the *generalised decision* (see p. 20), but translated into our proper setting. The subsequent Section 2.3 is bent on extending the previous ideas towards reflexive relations. Emphasis is put on the consequences of different semantics for the same syntax. The influence of this extension on information measures is the object of study of Section 2.4. Finally, Section 2.5 and Appendix 2.B discuss rough sets and decision trees from the point of view of the preceding sections. Appendix 2.A puts all classifiers into a perspective of partition-based reasoning.

To begin with, we let our attention swirl towards the notion of classification and the very straightforward portal between the real world and its mathematical counterpart.

2.2 Elementary granulation

2.2.1 Introduction

Notions and conventions. We denote the power set of a set X, i.e. the set of all subsets of X, by 2^X. The cardinality of a set X is denoted by $|X|$.

2^X

$|X|$

Given a function $g : X \to Y$, with finite Y, we can conceive of several ways to define $g(A)$ for a subset $A \subseteq X$. First we can focus on the set-characteristics of A:

$$g(A) = \begin{cases} \bigcup_{a \in A} \{g(a)\} & \text{, if } A \neq \emptyset\,, \\ \emptyset (\text{= unknown, no information}) & \text{, if } A = \emptyset\,. \end{cases} \tag{2.2.1}$$

SET INTERPRETATION.

We call this the **set interpretation**. Another track is to emphasise the distributional properties. If A is finite, we can observe the frequency distribution over Y:

$$\begin{array}{rcccl} g(A) & : & Y & \to & \mathbb{N}\,, \\ & & y & \mapsto & g(A, y) = |\{a \in A \mid g(a) = y\}|\,. \end{array} \tag{2.2.2}$$

If A is infinite (or finite), we can work with a conditional probability distribution (or an estimation of it) over Y:

$$\begin{array}{rcccl} g(A) & : & Y & \rightarrow & [0,1]\,, \\ & & y & \mapsto & g(A,y) = \mathcal{P}(y \mid A)\,, \end{array} \tag{2.2.3}$$

where $\mathcal{P}(y \mid A)$ stands short for $\mathcal{P}(g(a) = y \mid a \in A)$. This is the **distribution interpretation**.

DISTRIBUTION INTERPRETATION.

It should be clear that the first interpretation of $g(A)$ is included in the other two: indeed, taking the support of the frequency or probability distribution results back into (2.2.1). Therefore, if we refer to $g(A)$ as a set, we either refer to (2.2.1) or to the support of the other two interpretations. Other possibilities will not be dealt with in this chapter.

2.2.2 Structuring the objects

The real world and our perception.

clas·si·fy. *Transitive verb:* To arrange or organise according to class or category. – *The American Heritage® Dictionary of the English Language, Fourth Edition.*

This is how classification is perceived in everyday life. We tag the objects that attract our attention and say they belong to this or that class. We can easily formalise this labelling process as a function λ from some set of objects Ω, which we call the **object space**[1], to some set of labels $\mathcal{L}$:

OBJECT SPACE.

$$\lambda : \Omega \rightarrow \mathcal{L}\,.$$

The set of labels $\mathcal{L}$ is intangible and part of our conception of the objects under consideration. So, the classification provides us with a link between the real world and some perception of it.

If we assume $\mathcal{L}$ to be finite, λ is called a **classification**. Each label $\ell \in \mathcal{L}$ defines a **class** or **concept** $C_\ell = \lambda^{-1}(\ell)$ in Ω.

CLASSIFICATION.
CLASS.
CONCEPT.

Supervised learning. In the learning paradigm, we lack full awareness of the classification λ, and are only allowed a mere glimpse of it. We are only granted access to λ on a finite subset $\mathcal{S} \subseteq \Omega$, also referred to as the **sample space**. This limited view $\lambda_{|\mathcal{S}}$ will be denoted as $d : \mathcal{S} \rightarrow \mathcal{L}$, and is called the **decision function**. The couple $\Lambda = (\mathcal{S}, d)$ is called a **learning sample**. The whole intention is to induce the unknown λ as good as possible from $d = \lambda_{|\mathcal{S}}$. To accomplish such a feat, we obviously need some more structural knowledge about the objects of Ω.

SAMPLE SPACE.
DECISION FUNCTION.
LEARNING SAMPLE Λ.

[1]Other names can be found in the literature, such as *universe of discourse*

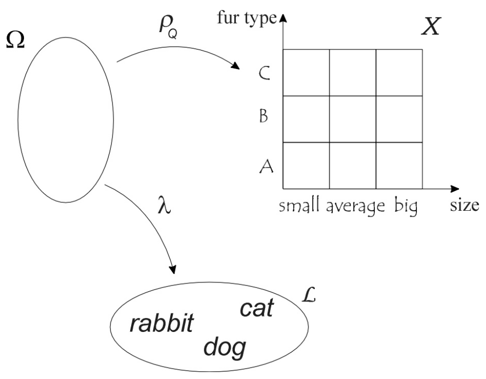


Figure 2.1: Elementary portals.

The mathematical world.

> **class.** *Noun:* A set, collection, group, or configuration containing members regarded as having certain attributes or traits in common; a kind or category. – *The American Heritage® Dictionary of the English Language, Fourth Edition.*

We cannot manipulate the objects from Ω themselves, but we can play with our knowledge about them. In our learning context, we suppose, as usual, that our primary information about the individuals in Ω is embodied by a finite set $Q = \{q_1, \ldots, q_n\}$ of descriptor variables (**attributes**) $q : \Omega \to \mathcal{X}_q$. Consequently, each object is intertwined with a specific vector within the **data space** $\mathcal{X} = \prod_{q \in Q} \mathcal{X}_q$ by the **descriptive representation**

ATTRIBUTES. DATA SPACE. DESCRIPTIVE REPRESENTATION.

$$\begin{array}{rcccl} \rho_Q & : & \Omega & \to & \Omega_{\mathcal{X}} \subseteq \mathcal{X}, \\ & & a & \mapsto & (q_1(a), \ldots, q_n(a)) \,. \end{array}$$

If the context leaves no doubt concerning the set Q, we simply write ρ instead of ρ_Q. To keep notations manageable, we sometimes write the representation of an object $a \in \Omega$ in the data space $\mathcal{X}$ as $\mathbf{a}$, in other words $\rho(a) = \mathbf{a} \in \mathcal{X}$. If we mention elements of $\mathcal{X}$ without reference to an object of Ω, we use boldface characters $\mathbf{x}, \mathbf{y}, \ldots$

This simple function ρ creates a gateway between the real world and a mathematical representation of it. In doing so, the objects of Ω are grouped into **elementary (information) granules**: each vector $\mathbf{x}$ of $\mathcal{X}$ wraps the set of objects $\rho^{-1}(\mathbf{x})$ (if $\mathbf{x} \notin \Omega_{\mathcal{X}}$, then $\rho^{-1}(\mathbf{x}) = \emptyset$).

ELEMENTARY (INFORMATION) GRANULES.

Up to this point we have just formalised the typical information that is contained in a basic entry for supervised learning.

2.2.3 Elementary representation of the decision function

The next step is to create a mathematical counterpart of the classification or decision function, where the domain is now the data space $\mathcal{X}$ rather than the object space Ω. We don't have many options to pull this off, with the limited building blocks we dispose of. Since we want to transfer the information of λ from Ω to $\mathcal{X}$, we have to rely on our only link between these two spaces, namely the descriptive representation.

Definition 2.2.1

- The **elementary representation of a classification** λ (w.r.t. to a fixed finite set of attributes) is defined by

ELEMENTARY REPRESENTATION OF A CLASSIFICATION.

$$\boxed{\hat{\lambda} = \lambda \circ \rho^{-1}}$$

- The **elementary representation of a decision function** $d = \lambda_{|\mathcal{S}}$ is defined by

ELEMENTARY REPRESENTATION OF A DECISION FUNCTION.

$$\boxed{\hat{d} = d \circ \rho_{|\mathcal{S}}^{-1}}$$

It should be clear that although $d = \lambda_{|\mathcal{S}}$, we may have that $\hat{d} \neq \hat{\lambda}_{|\rho(\mathcal{S})}$. Let us make these definitions more concrete. For all $\mathbf{x} \in \mathcal{X}$, we have $\hat{\lambda}(\mathbf{x}) = \lambda(\rho^{-1}(\mathbf{x}))$. But since $\rho^{-1}(\mathbf{x}) \subseteq \mathcal{X}$, we already need to make a choice between (2.2.1), (2.2.2) and (2.2.3).

Representation based on sets. Applying the set interpretation (2.2.1), we arrive at

$$\begin{aligned} \hat{\lambda}(\mathbf{x}) &= \lambda(\rho^{-1}(\mathbf{x})) = \bigcup_{a \in \rho^{-1}(\mathbf{x})} \{f(a)\} \\ &= \{\ell \in \mathcal{L} \mid (\exists a \in \Omega)(\rho(a) = \mathbf{x} \wedge \lambda(a) = \ell)\}\,. \end{aligned}$$

Note that this means that $\hat{\lambda} : \mathcal{X} \to 2^{\mathcal{L}}$. Thus, the representation of a classification is again a classification, but now in the space $\mathcal{X}$, with classes $\hat{\lambda}^{-1}(L)$, where $L \in 2^{\mathcal{L}}$. Moreover, if the object space and the data space are isomorphic, $\Omega \cong \mathcal{X}$, then $\hat{\lambda} \cong \lambda$. The sets $\mathbf{C}_L := \rho^{-1}(\hat{\lambda}^{-1}(L)) \subseteq \Omega$ (we write $\mathbf{C}_{\{\ell\}}$ shortly as $\mathbf{C}_\ell$) are sometimes called the **decision regions**. They constitute the view on the classes of Ω as seen through the mathematical model. It is clear that a minimal condition for the existence of doubt between a and b is that they belong to the same decision region derived from a set $L \subseteq \mathcal{L}$ with $|L| > 1$.

DECISION REGIONS.

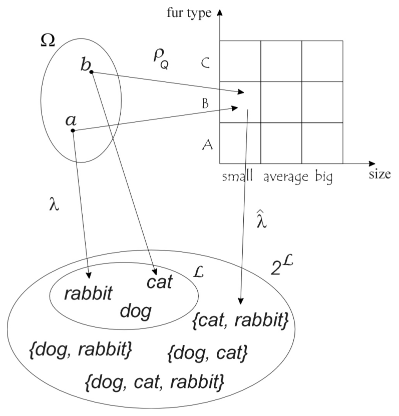


Figure 2.2: Doubt introduced by modelling λ.

DOUBT. This representation introduces the notion of **doubt**[2] between objects $a, b \in \Omega$ whenever $\rho(a) = \rho(b)$, but $\lambda(a) \neq \lambda(b)$. In that case, it holds that $|\hat{\lambda}(\rho(a))| > 1$. We only consider doubt that arises by the modelling of λ as depicted in Figure 2.2. Similarly, we find for the representation of $d = \lambda_{|\mathcal{S}}$ that

$$\begin{aligned} \hat{d}(\mathbf{x}) &= d(\rho_{|\mathcal{S}}^{-1}(\mathbf{x})) \\ &= \{\ell \in \mathcal{L} \mid (\exists a \in \mathcal{S})(\rho(a) = \mathbf{x} \wedge \lambda(a) = \ell)\} . \end{aligned}$$

Representation based on distributions. If we narrow our view only to sets, we neglect quite some of the available information. While this might be interesting in order not to drown in too many details and retain the general impressionistic picture, it may not be sufficient in other situations. In the latter case, we can add detail by using (2.2.2), to keep track of the number of objects within one information granule that are mapped onto each label, or (2.2.3) to retain the relative frequencies.

Returning to the objects. If we want to apply the previous models (representations) directly to an object $a \in \Omega$, we first need to transform it into its descriptive representation $\rho(a)$. So, obviously, there is nothing to prevent us from deriving the twin function $\hat{\lambda}^*$ on Ω through ρ as $\hat{\lambda}^* = \hat{\lambda} \circ \rho$, i.e. for $a \in \Omega$

$$\hat{\lambda}^*(a) = \hat{\lambda}(\mathbf{a}) .$$

Within the limitations imposed by our representation of Ω by $\mathcal{X}$ and our choice of interpretation of $\lambda(A)$ for $A \subseteq \Omega$, the model $\hat{\lambda}^*$ is the best approximation of $\lambda : \Omega \to \mathcal{L}$ we may possibly conceive of.

[2]The literature harbours other terminology indicating the same phenomenon, such as *inconsistency* (indiscernible objects with different labels). In Chapter 4 it will become clear why we opted for a different name.

Remark that, following the representation based on sets, the decision regions can be written in function of $\hat{\lambda}^*$ as follows: $\mathbf{C}_L = \rho^{-1}(\hat{\lambda}^{-1}(L)) = (\hat{\lambda}^*)^{-1}(L)$.
We can also derive such a twin function[3] $\hat{d}^*$ for d. Remark how this simple extension has already initiated the learning process: in contrast with d which is restricted to the finite set of samples $\mathcal{S} \subseteq \Omega$, its twin $\hat{d}^*$ is defined on whole Ω (even if it may still map a lot of objects to the empty set). Remark that while $\hat{d}^*$ becomes the best operational approximation of d, it is not necessarily a good approximation of λ.

2.2.4 Summary

- The objects $a \in \Omega$ are represented by their description $\rho_Q(a) \in \mathcal{X}$, inducing an elementary granulation of Ω.
- Using either the set or the distribution interpretation, the classification $\lambda : \Omega \to \mathcal{L}$ can be represented by the mapping $\hat{\lambda}$ on $\mathcal{X}$:
$$\hat{\lambda} = \lambda \circ \rho^{-1} \,.$$
- *Doubt* arises if two objects have the same elementary representation according to Q, but belong to different classes according to λ:
$$\rho_Q(a) = \rho_Q(b) \quad \text{and} \quad \lambda(a) \neq \lambda(b) \,.$$

2.3 Relational granulation

Notions and conventions. If we speak about a relation $R \subseteq X \times Y$, we always refer to a binary relation, and denote $R(a) = \{b \in Y \mid aRb\}$. In this way, R can be seen as a mapping from X to 2^Y, and can therefore be extended towards a mapping from 2^X to 2^Y in the way of (2.2.1), i.e. $R(a)$

$$R(A) = \bigcup_{a \in A} R(a) = \{b \in Y \mid (\exists a \in A)(aRb)\} \,.$$

Also, the relational composition $R_1 \circ R_2$ has to be read as "R_1 follows R_2". However, if we speak of the inverse R^{-1} of a relation, we do not interpret it as a function, but simply mean that $bR^{-1}a \iff aRb$. With a relation R on X, we mean $R \subseteq X \times X$. A relation R is said to be embedded in another relation R' if $aRb \Rightarrow aR'b$, or equivalently, if for all $a \in X$ it holds that $R(a) \subseteq R'(a)$. $R_1 \circ R_2$ R^{-1}
A **partition** Π of a set X is a set of non-empty, pairwise disjoint subsets π of X such that $\bigcup_{\pi \in \Pi} \pi = X$. The elements π of a partition Π are called **blocks**. Parti- PARTITION. BLOCKS.

[3]This function is also known as the *generalised decision* (see p. 20).

EQUIVALENCE RELATIONS. tions on X stand in one-to-one correspondence with **equivalence relations** on X (relations that are reflexive, symmetric and transitive). The blocks correspond to the equivalence classes. Usually, we skip back and forth between partitions and equivalence relations without changing the notation, e.g. we take a partition Π on X and then consider it as a mapping: $X \to 2^X$.

SIMILARITY RELATION. There exist quite a number of different definitions for a **similarity relation** S in the literature, referring to mathematically different concepts. Still, all these definitions have at least two things in common: a syntax demanding (at least) reflexivity, and a semantics demanding that aSb can be meaningfully read as "a is similar to/is like b". We take this largest common divisor as the definition of a similarity relation[4].

2.3.1 Introduction

Without any effort, some primal learning has already been achieved through the function $\hat{d}^*$. The granularity introduced in Ω by its mathematical representation $\mathcal{X}$ lies at the very heart of this matter. Indeed, if for $a \in \Omega$ it holds that $\rho^{-1}(\rho(a)) \cap \mathcal{S} \neq \emptyset$, then $\hat{d}^*(a) \neq \emptyset$. In human language this translates to: if we have one (or more) sample object(s) with the same description as the object a, we can say something meaningful about the classification of the object a. The granule around an object a functions as a kind of recipient of information that can be applied to a. This brings us to the idea of enlarging the granules in Ω by relating more objects to each other. The most intuitive and natural way to achieve this, would be to relate resembling objects to one another, objects that share specific characteristics, or have similar characteristics. Since in the present context[5], the objects' characteristics are captured by their vector representation in $\mathcal{X}$, we will consider the impact of incorporating similarity relations on $\mathcal{X}$ into the representations.

2.3.2 Restructuring the objects

The objects are initially structured by the descriptive representation ρ. The inverse ρ^{-1} induces an equivalence relation on Ω, corresponding with the minimal reflexive relation $\mathbf{1}$ that can be defined on $\mathcal{X}$: we have $\mathbf{x}\mathbf{1}\mathbf{y}$ if and only if $\mathbf{x} = \mathbf{y}$. So, we can rewrite an elementary granule of Ω as the result of the mapping $\rho^{-1} \circ \mathbf{1} \circ \rho$. Consider any equivalence relation (partition) Π on Ω. All objects inside one block are then considered equivalent, and therefore treated the same. Hence, instead of keeping the focus on the individual objects, we could simply classify the blocks $\pi \in \Pi$, using either a set or distribution interpretation for $\hat{\lambda}(\pi)$. Afterwards we can state that an object a is classified the same as $\Pi(a)$.

A more general way of achieving the same is noticing that $\mathbf{1} \subseteq \Pi$, where Π now stands for an equivalence relation on $\mathcal{X}$. Without any difficulties we can now broaden the elementary granules into new – and possibly more informative – granules delimited by $\rho^{-1} \circ \Pi \circ \rho$. These granules will still partition the object space Ω.

[4]In many papers, also symmetry is required in the syntax, but in [114], a discussion can be found about abandoning the symmetry requirement.

[5]We do not mingle with relational data bases.

However, we do not have to limit ourselves to equivalence relations, any reflexive relation R will do, as long as we can interpret (as for example in the case of a similarity relation, see below) the resulting granules

$$\boxed{\rho^{-1} \circ R \circ \rho}$$

We denote the granule around $a \in \Omega$ as $[a]_R = \rho^{-1}(R(\rho(a)))$.

$[a]_R$

For example, if two objects belong to the same granule w.r.t. to some equivalence relation Π, then these two objects may be considered equivalent w.r.t. to Π. However, we may also consider a similarity relation S, and investigate the granules $\rho^{-1} \circ S \circ \rho$. Now the granules may intersect each other and more attention must be paid to their interpretation. Indeed, the fact that a and b belong to the same granule does not mean they are similar, since there is not necessarily symmetry and transitivity. We come back to this in the following section.

2.3.3 Relational representation of the decision function

We may now apply λ and d on the new granules derived from a reflexive relation R and define the **relational representation of a classification** λ as (see also Figure 2.3):

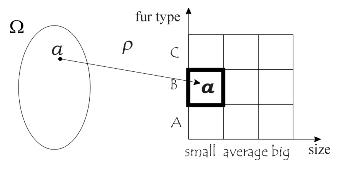
RELATIONAL REPRESENTATION OF A CLASSIFICATION. $\hat{\lambda}^*_R$

$$\hat{\lambda}^*_R = \lambda \circ \rho^{-1} \circ R \circ \rho\,, \tag{2.3.1}$$

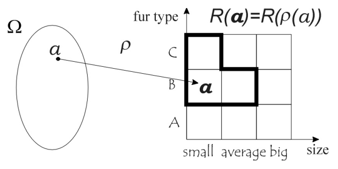
and the **relational representation of a decision function** d as:

RELATIONAL REPRESENTATION OF A DECISION FUNCTION. $\hat{d}^*_R$

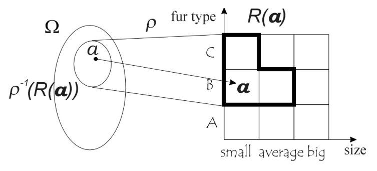
$$\hat{d}^*_R = d \circ \rho^{-1}_{|\mathcal{S}} \circ R \circ \rho\,. \tag{2.3.2}$$



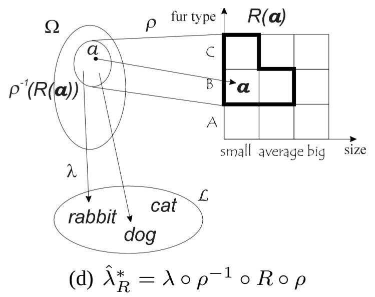
(a) ρ



(b) $R \circ \rho$



(c) $\rho^{-1} \circ R \circ \rho$



(d) $\hat{\lambda}^*_R = \lambda \circ \rho^{-1} \circ R \circ \rho$

Figure 2.3: Relational representation.

Within the set representation, the notions of doubt and decision regions are readily extended towards the context of partitions. There is **doubt** between objects a and b when $R(\rho(a)) = R(\rho(b))$, but $f(a) \neq f(b)$, in which case $\hat{\lambda}^*_R(a) = \hat{\lambda}^*_R(b)$ and $|\hat{\lambda}^*_R(a)| > 1$. Decision regions are just the sets $(\hat{\lambda}^*_R)^{-1}(L)$ with $L \subseteq 2^{\mathcal{L}}$.

Lemma 2.3.1. *A sufficient condition for $\hat{d}^*_R : \Omega \to 2^{\mathcal{L}}$ never to attain the value $\emptyset$ is*

$$(\forall \mathbf{x} \in \mathcal{X})(R(\mathbf{x}) \cap \rho(\Omega) \neq \emptyset \Rightarrow R(\mathbf{x}) \cap \rho(\mathcal{S}) \neq \emptyset)\,.$$

In many cases, it is assumed that $\rho(\Omega) = \mathcal{X}$, which simplifies the previous condition to $(\forall \mathbf{x} \in \mathcal{X})(R(\mathbf{x}) \cap \rho(\mathcal{S}) \neq \emptyset)$. If R is an equivalence relation (partition) Π, we obtain the condition:

$$(\forall \pi \in \Pi)(\pi \cap \rho(\mathcal{S}) \neq \emptyset)\,.$$

Decision trees typically investigate partitions for which this condition holds.

Interpretation of granules. As can be seen, the information for a now comes from all objects related to it. Using equivalence relations just means that

(i) reflexivity: each object a is informative for itself,

(ii) symmetry: if the information on a can be used for b, then the information on b can also be used for a,

(iii) transitivity: if a is considered informative for b, and b for c, than a is also considered informative for c.

Obviously, of these three properties, only reflexivity is a conditio sine qua non. This means that any relation that may seem interesting can be used, as long as the reflexive closure[6] is considered.
Similarity relations S are a special kind of reflexive relations. Their only non-discussable mathematical requirement is reflexivity. However, in addition they also need to follow some semantics (i.e. not every reflexive relation is a similarity relation): $\mathbf{x}S\mathbf{y}$ should mean that $\mathbf{x}$ is similar to $\mathbf{y}$. In this way, a granule $[a]_S$ is readily interpreted as the set of objects a resembles to. Therefore it is meaningful to use the information of the objects in $[a]_S$ as additional information for a. On the other hand, a is not necessarily informative to the other objects in $[a]_S$: a simple situation in a small firm may resemble a complex situation in a large firm and therefore the small firm can benefit from how the situation was handled in the large firm, the other way around is however less obvious. As a consequence, it is not useful to consider granules based on S^{-1} for this interpretation of S.
To clarify the previous even more, let $\mathcal{X}$ consist of one single axis derived from the attribute "size". Now define S as: $\mathbf{x}S\mathbf{y}$ if $|\mathbf{x} - \mathbf{y}| \leq 10\%$ of $\mathbf{y}$, or translated to the objects: a is related to b if $|\text{size}(a) - \text{size}(b)| \leq 10\%$ of size(b). Now we can interpret this relation in two different ways:

[6]The *reflexive closure of a relation R on X is the minimal reflexive relation R' on X that contains R. So, $aR'a$ for all $a \in X$ and $aR'b$ (with $a \neq b$) if and only if aRb.*

(i) In the first case, S represents the fact that there may exist an error of 10% on the measurements. Since S is defined relatively w.r.t. its second argument, we have that for a given $\mathbf{x}$, the set of values $\mathbf{y}$ within a 10% error bound w.r.t. $\mathbf{x}$ is given by $S^{-1}(\mathbf{x})$, the set of all $\mathbf{y}$ such that $|\mathbf{x} - \mathbf{y}| \leq 10\%$ of $\mathbf{y}$. However, when observing $\mathbf{x}$, we are only interested in $S(\mathbf{x})$: assume we observe $\mathbf{x}$ and that the correct value is in fact $\mathbf{y}$, since the observation $\mathbf{x}$ may differ up to 10% from the actual value $\mathbf{y}$, we have $\mathbf{x}S\mathbf{y}$. This is a so-called *disjunctive* interpretation: we observe $\rho(a) = \mathbf{a}$ for some object a, but are only certain that the (unique) correct measurement of a belongs to one of the values in $S(\mathbf{a})$. In fact, the role of S is that of an **imprecision relation** relation [38], and not of a similarity relation.

IMPRECISION RELATION.

(ii) In the second case, the measurements are precise (or at least assumed to be), but we want to take into account the resemblances between objects. For example, we are capable of perfectly measuring the size of objects, yet we have reasons to state that distinguishing an object a from another object b is meaningless if the size of a is not at least 10% smaller or larger than the size of b, i.e. if $\mathbf{b} \in S(\mathbf{a}) = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{a}S\mathbf{x}\}$. This is a *conjunctive* interpretation: from the viewpoint of $\mathbf{x}$, no distinction is made between any of the vectors in $S(\mathbf{x})$. Here S fulfills its role of a similarity relation.

From this example it becomes clear that applying the functions (2.3.1) or (2.3.2) without considering the interpretation of R can be very dangerous. Indeed, in the previous example, if S is seen as a similarity relation, we implicitly agree on the fact that all the objects in $[a]_S$ add useful information to the classification of a (why should we otherwise bother considering similarity relations in this context?). On the other hand, interpreting S as an imprecision relation does not imply such an agreement, therefore compromising the use of the proposed relational representations. Indeed, knowing that one of the measurements in $S(\mathbf{a})$ is the correct one, only implies that some of the objects in $[a]_S$ are useful for the classification of a. When deriving expressions for measuring the amount of information a granule contains, this distinction will again play an important role (see Section 2.4).
If all the objects in a granule around an object a impart their information on a, we call this granule an **information granule**. Although we do not consider it, it is interesting to remark that graded/fuzzy relations can lead to a kind of graded/fuzzy information granules.

INFORMATION GRANULE.

Combining relations. So far, we have focussed on two kinds of relations for enlarging the elementary granules: equivalence relations and similarity relations. They differ substantially from each other in that the former only needs to satisfy a specific syntax to result in information granules, while the latter are dependent of an additional semantics. As a consequence, the equivalence relations can be produced by computer, but similarity relations must be furnished by human interaction.
Now the question arises how we can deal with problems that combine both kind of relations? For example to keep the smooth running machinery of decision trees, but

with the conviction that we should not distinguish between measurements within a certain neighbourhood[7], which can be modelled by imposing a similarity relation on $\mathcal{X}$.

As soon as you realise that any relation on the data space $\mathcal{X}$ takes the finest equivalence relation $\mathbf{1}$ into consideration, the solution starts dawning. Each granule $R(\mathbf{x})$ (in $\mathcal{X}$) is "centered" around an equivalence class of $\mathbf{1}$. The granule $R(\mathbf{x})$ arises as if by looking at this equivalence class (that contains but $\mathbf{x}$) through a set of glasses covered with some R-coating: while scrutinising the elements of the equivalence class, we see some additional vectors in the corner of our eyes, namely the vectors $\mathbf{y}$ such that $\mathbf{x}R\mathbf{y}$. The same idea applies for larger equivalence classes π: a granule is then centered around a set π of vectors, and covers the R-vision of π. Since we can behold just one vector at a time using R, we let our eye skim over π vector by vector:

$$R(\pi) = \bigcup_{\mathbf{x}\in\pi} R(\mathbf{x}) = \{\mathbf{z} \in \mathcal{X} \mid (\exists \mathbf{x} \in \pi)(\mathbf{x}R\mathbf{z})\} .$$

Hence, the resulting granules originate from the composition $R \circ \Pi$:

$$\mathbf{x}(R \circ \Pi)\mathbf{z} \iff (\exists \mathbf{y} \in \mathcal{X})(\mathbf{x}\Pi\mathbf{y} \wedge \mathbf{y}R\mathbf{z}) .$$

Since $\mathbf{z}$ affects $\mathbf{y}$, and $\mathbf{y}$ and $\mathbf{x}$ are treated the same, $\mathbf{z}$ also affects $\mathbf{x}$.

As before we can apply λ and d on these new granules, or, amounting to the same, use Equations (2.3.1) and (2.3.2) with $R \circ \Pi$ as reflexive relation.

2.3.4 Summary

- The elementary granulation of Ω can be coarsened by a reflexive relation R.
- The resulting granules $[a]_R = \rho^{-1}(R(\rho(a)))$ are called information granules if the information about the objects inside the granule $[a]_R$ can be used as information about a.
- If R leads to information granules, either the set or the distribution interpretation can be used to represent the classification $\lambda : \Omega \to \mathcal{L}$ by the mapping $\hat{\lambda}^*_R$ on Ω:
$$\hat{\lambda}^*_R = \lambda \circ \rho^{-1} \circ R \circ \rho .$$

[7]Mark the distinction with measurement noise which can be handled by assigning an object to an equivalence class with a certain probability [93].

2.4 Corollary 1: Impurity measures in supervised learning

2.4.1 Introduction

A next step is to measure the information in the presence of all these different relations. In the case we only have the descriptive representation or, more generally, when confronted with an equivalence relation, this problem has several known solutions based on impurity measures [23]. An **impurity measure** (see p. 16) is a non-negative function ϕ_n from the set of probability distributions over the class labels $\mathcal{L} = \{1, \ldots, k\}$ to $\mathbb{R}$ such that

IMPURITY MEASURE.

- ϕ_n reaches its maximum in $(\frac{1}{n}, \ldots, \frac{1}{n})$;
- ϕ_n is zero if there is no uncertainty:

$$\phi_n(1, 0, \ldots, 0) = \phi_n(0, 1, 0, \ldots, 0) = \phi_n(0, \ldots, 0, 1) = 0\,;$$

- ϕ_n is symmetric: for all permutations σ of $\{1, \ldots, n\}$, we have

$$\phi_n(p_1, \ldots, p_n) = \phi_n(p_{\sigma(1)}, \ldots, p_{\sigma(n)})\,.$$

In order to obtain a higher discrimination power with dropping uncertainty, also the condition of strict concavity must be added:

$$\phi_n''(p_1, \ldots, p_n) < 0\,.$$

The most frequently used impurity measures, the Shannon entropy [101] and the Gini diversity index, oblige this last condition. This section deals in depth with these two measures, which will enable us to extend their scope quite easily in the next section.

2.4.2 Some specific impurity measures

The Hartley information. The Shannon entropy finds its origin in the Hartley information [55]. Hartley indicated that when one element is chosen from a finite set S of equally likely choices then the number of possible choices or any monotonic function of this number can be regarded as a measure of information. He additionally pointed out that the logarithmic function is the most "natural" measure, and so he defined $I(S) := \log_2(|S|)$. The conditional Hartley information $I(U|S)$, where $U \subseteq S$, boils down to the comparison of the a priori information $I(S)$ with the a posteriori information $I(U)$. Hence,

$$I(U|S) := I(S) - I(U) = \log_2\left(\frac{|S|}{|U|}\right) = -\log_2\left(\frac{|U|}{|S|}\right)\,.$$

The Shannon entropy. In a probabilistic setting, where the probability of choosing $s \in S$ is defined as $\mathcal{P}(s) = 1/|S|$, the Hartley information represents the information we have if the state of a random variable X (taking values in S) is known, or the uncertainty if it is unknown. More general, let X be a random variable with a finite domain $\mathrm{dom}(X)$, and let $\mathcal{P}(X = x)$ denote the probability that X takes on the value x. If the $\mathcal{P}(X = x)$ are rational numbers[8], we can think of the following experiment: a blind pick of an element of a set S containing for each $x \in \mathrm{dom}(X)$ a proportion $\mathcal{P}(X = x)$ of elements with label x. After the execution of this experiment, we hold an element with label x, so we posses the information $I(U_x|S)$, where $U_x \subseteq S$ corresponds to the subset of elements with label x. In this experiment, the probability of grabbing an element of U_x is $\mathcal{P}(X = x) = |U|/|S|$, and the information we wield when $X = x$ is defined as $H(X = x) = I(U_x|S) = -\log_2 \mathcal{P}(X = x)$. (also known as the Wiener entropy [123]). The uniqueness of this function can be demonstrated under the condition that it is non-negativity, additive and normalised (see [1]). The probability of obtaining this information in such an experiment (i.e. the probability of stumbling upon an element of U_x) is $\mathcal{P}(X=x)$. Each time we repeat this experiment, it results in some information $H(X = x)$. The information of the random variable X is then defined as the average information resulting from a draw, in other words, the weighted arithmetical mean

$$H(X) = \sum_{x\in\mathrm{dom}(X)} \mathcal{P}(X=x)\cdot H(X=x) = -\sum_{x\in\mathrm{dom}(X)} \mathcal{P}(x)\cdot\log_2 \mathcal{P}(x)\,,$$

SHANNON ENTROPY. CONDITIONAL SHANNON ENTROPY.

where $P(x) = P(X = x)$ and $0\log_2 0 := 0$. This expression is nothing else but the **Shannon entropy** (plenty of characterisations can be found in [1]). The **conditional Shannon entropy** of a variable Y if the variable X is known, is also founded on the same principle:

$$\begin{aligned} H(Y|X) &= \sum_{x\in\mathrm{dom}(X)} \mathcal{P}(X=x)\cdot H(Y|X=x) \\ &= \sum_{x\in\mathrm{dom}(X)} \mathcal{P}(x)\cdot\left(-\sum_{y\in\mathrm{dom}(Y)} \mathcal{P}(y|x)\log_2 \mathcal{P}(y|x)\right)\,, \end{aligned}$$

where $\mathcal{P}(y|x) = \mathcal{P}(Y = y|X = x)$ represents the conditional probability conditionelle that $Y = y$ if we known that the variable X takes the value x. In fact, it is possible to define $H(Y|X)$ starting from the natural condition $H_{mn}(X,Y) = H_m(X) + H_n(Y|X)$, where $|\mathrm{dom}|\,(X) = m$ and $|\mathrm{dom}|\,(Y) = n$ ([1]): the information expected from two experiments corresponds to the information expected from the first experiment plus the conditional information of the second experiment w.r.t. the first one.

[8]When the probabilities are real numbers, one has to consider the non-atomic Kolmogorov algebras (see [94]).

The Shannon entropy in supervised learning. The information that carries away our interest is the one linked with the classification λ. In the lottery experiment, the elements (objects) are then labelled by λ. More particular, we are inquisitive about the the information contained in the random variable Y with $\mathrm{dom}(Y) = \mathcal{L}$, and this in the presence of the information contained in the equivalence relation[9] Π. The latter corresponds to the information from the random variable X with $\mathrm{dom}(X) = \Pi$ (picking an object from Ω labelled by $\Pi \circ \rho$). In other words, there are two experiments in the run: first of all the drawing of an object from Ω to determine the equivalence class $\pi \in \Pi$, and next the (conditional) drawing of an object from π to determine the label in $\mathcal{L}$. Together, this amounts to

$$H(Y|X) = \sum_{\pi\in\Pi} \mathcal{P}(\pi) \cdot \left(- \sum_{i\in\mathcal{L}} \mathcal{P}(i|\pi) \log_2 P(i|\pi) \right) . \tag{2.4.1}$$

Remark that $\mathcal{P}(i|\pi) = \hat{\lambda}(\pi, i)$, if we follow (2.2.3).
In supervised learning, we only dispose of estimates p (obtained from Λ) of the probabilities $\mathcal{P}$ needed in this formula. Plugging these estimates into this formula leads to the information contained in the learning sample Λ, denoted by $H(\Lambda|\Pi)$.

The Gini diversity index. The origins of the diversity index can be traced back to statistics, and more specifically to the analysis of variances. For each block π of Π, it is defined as

$$G(\Lambda|\pi) = \sum_{\substack{(i,j)\in\mathcal{L}^2 \\ i\neq j}} p(i|\pi)\, p(j|\pi) = 1 - \sum_{i\in\mathcal{L}} p^2(i|\pi) \, .$$

This expression can be interpreted as a variance of dichotomous classifications, or still, as the probability of assigning a wrong class label to an object of π when this is done by assigning it the label of an object drawn from the set $\rho^{-1}(\pi)$. As was the case for the Shannon entropy, the information w.r.t. the partition Π is again a probabilistic average:

$$G(\Lambda|\Pi) = \sum_{\pi\in\Pi} p(\pi) \cdot G(\Lambda|\pi) \, . \tag{2.4.2}$$

This expression can also be interpreted as the (estimated) probability of attaching an incorrect class label to an object of Ω in the presence of the relation Π.

2.4.3 Information in the presence of a similarity relation

Introduction. To construct an information measure that incorporates a similarity relation, we only need to gain an understanding of how this relation interacts with the probabilities encountered in the expressions (2.4.1) and (2.4.2).
Concerning imprecision relation, this problem has been discussed in [59]. Next, we tackle the problem of including a similarity relation.

[9]If Π is the identity relation, then $\Pi \cong \mathcal{X}$ using the isomorphism $\{\mathbf{x}\} \leftrightarrow \mathbf{x}$.

The Shannon entropy. First, we rewrite the expression (2.4.1) in its most elementary form, i.e. for the special case $\Pi = \mathbf{1}$:

$$H(\Lambda|\mathbf{1}) := \sum_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}) \cdot H(\Lambda|\,\mathbf{x})\,. \tag{2.4.3}$$

Now imagine again the blind picking of an object from Ω. This object determines a block $\rho(a) = \mathbf{a}$ in $\mathcal{X}$, and the (estimated) information about the classification linked to this block $\mathbf{a}$ is just $H(\Lambda|\,\mathbf{a})$. The previous formula conveys the average information we receive from such an experiment about the classification (w.r.t. the identity relation).

Now, the only thing we want to do, is to express the average information obtained from such an experiment in the presence of Π and a similarity relation S. This situation differs from the previous one only in the information connected to the granules in $\mathcal{X}$. In the presence of S, we agree on the principle that all objects of $[a]_S$ deliver useful information about the classification of objects represented by $\mathbf{x}$. Indeed, this constitutes the foundation of our reasoning during the construction of the relational representation $\hat{d}^*_S$ of λ. Hence, the information linked to the granule $\mathbf{x}$ becomes $H(\Lambda|S(\mathbf{x}))$. We emphasise on the fact that we did not change anything in our experimental settings, and therefore also the associated probabilities remain unchanged. The expression (2.4.3) is transformed into

$$H(\Lambda|S) = H(\Lambda|S\circ\mathbf{1}) := \sum_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}) \cdot H(\Lambda|S(\mathbf{x}))\,.$$

It is now easy to conceive of the modifications needed to deal with the combined presence of $S\circ\Pi$:

$$\begin{aligned} H(\Lambda|S\circ\Pi) &:= \sum_{\pi\in\Pi} p(\pi)\cdot H(\Lambda|S(\pi)) \\ &= \sum_{\pi\in\Pi} p(\pi)\cdot\left(-\sum_{i\in\mathcal{L}} p(i|S(\pi))\log_2 p(i|S(\pi))\right). \end{aligned}$$

The Gini diversity index. We can treat the Gini diversity index in the same way, which results in

$$G(\Lambda|S\circ\Pi) = \sum_{\pi\in\mathcal{X}_\Pi} p(\pi)\cdot G(\Lambda|S(\pi))\,.$$

Moreover, this expression keeps its probabilistic interpretation of assigning an incorrect class label to an object of Ω when considering both Π and S. Indeed, the original labelling rule consisted of a drawing from the set $\rho^{-1}(\pi)$ to obtain the label. In the present context, the information comes from all objects of $\rho^{-1}(S(\pi))$, and not just from the objects of $\rho^{-1}(\pi)$.

2.5 Corollary 2: A partial reformulation of rough set methodology

2.5.1 Introduction

In this section, we approach the rough set methodology from a slightly different perspective than usual, shifting the center of gravity away from the notions of lower and upper approximation towards decision regions and doubt.
Rough set theory starts with narrowing its focus to the sample space $\mathcal{S}$ and the decision function d. No reference is made to Ω and λ. This is due to the origin of the theory in Information Systems [84] where the main goal was to represent and store data efficiently for easy retrieval. The idea of using rough sets in learning was only initiated in a second stage. Therefore, we may as well proceed our discussion with the assumption that $\mathcal{S} = \Omega$ (finite!), and consequently that $d = \lambda$.

2.5.2 Lower and upper approximations

Notions and conventions. For any subset$A \subseteq X$, the **characteristic function** $\chi_A : X \to \{0,1\}$ is defined by

CHARACTERISTIC FUNCTION χ_A.

$$\chi_A(a) = \begin{cases} 1 & \text{, if } a \in A\,, \\ 0 & \text{, if } a \notin A\,. \end{cases}$$

Let R be any relation on Ω, and $A \subseteq \Omega$. In "standard" rough set terminology, the *lower approximation* (see p. 19) of A is defined as

$$\underline{A}_R = \bigcup \{a \in \mathcal{S} \mid R(a) \subseteq A\}\,,$$

and the *upper approximation* (see p. 19) of A as

$$\overline{A}^R = \bigcup \{a \in \mathcal{S} \mid R(a) \cap A \neq \emptyset\}\,.$$

Granules and relations. In the previous sections, we always considered relations R on $\mathcal{X}$ and granules $[a]_R = \rho^{-1} \circ R \circ \rho(a)$. However, the rough set literature will always refer directly to relations R on Ω. A granule $[a]_R$ is then defined immediately as $R(a)$. To stay closer to the rough set literature, we will also formulate the propositions initially using relations defined directly on Ω.

"Standard" approximations. This section focusses on the lower and upper approximations of classes $C_\ell \subseteq \Omega$ derived from a classification $\lambda : \Omega \to \mathcal{L}$, i.e. $C_\ell = \lambda^{-1}(\ell) = \{a \in \Omega \mid \lambda(a) = \ell\}$ with $\ell \in \mathcal{L}$. However, all results can be rewritten for arbitrary subsets $A \subseteq \Omega$ instead of classes C_ℓ. Just replace λ by the characteristic function χ_A, and the corresponding classes C_1 and C_0 by A and the complement of A respectively.

The following proposition shows that the standard rough set methodology is based on the set interpretation (2.2.1) of the extension of a function $g : X \to Y$ to $g : 2^X \to Y$.

Proposition 2.5.1. *Let $\lambda : \Omega \to \mathcal{L}$ be a classification and let R be a reflexive relation on Ω. The lower approximation $\underline{C_\ell}_R$ and upper approximation $\overline{C_\ell}^R$ of C_ℓ w.r.t. R are characterised by*

$$a \in \underline{C_\ell}_R \iff \lambda_R(a) = \{\ell\} \qquad \textit{and} \qquad a \in \overline{C_\ell}^R \iff \ell \in \lambda_R(a)\,,$$

where $\lambda_R := \lambda \circ R$.

Proof.
Define the set $A = \{a \in \Omega \mid \lambda_R(a) = \{\ell\}\}$. We have

$$\begin{aligned} a \in A &\iff \lambda(R(a)) = \{\ell\} \\ &\iff (\forall b \in \Omega)(b \in R(a) \Rightarrow \lambda(b) = \ell) \\ &\iff R(a) \subseteq f^{-1}(\ell) = C_\ell\,, \end{aligned}$$

and thus, $A = \{a \in \Omega \mid R(a) \subseteq C_\ell\}$. The information granule around a is just $R(a)$, whence $R(a) = [a]_R$. This means that $A = \underline{C_\ell}_R$.
Now define the set $B = \{a \in \Omega \mid \ell \in \lambda_R(a)\}$. We have

$$\begin{aligned} a \in B &\iff (\exists b \in \Omega)(b \in R(a) \wedge \lambda(b) = \ell\} \\ &\iff R(a) \cap C_\ell \neq \emptyset\,, \end{aligned}$$

and thus, $B = \overline{C_\ell}^R$. □

Remark that, for a relation R on Ω

$$\begin{aligned} a \in \overline{C_\ell}^R &\iff (\exists b \in \Omega)(b \in R(a) \wedge f(b) = \ell\} \\ &\iff (\exists b \in \Omega)(a \in R^{-1}(b) \wedge b \in C_\ell\}\,, \end{aligned}$$

which implies $\overline{C_\ell}^R = \bigcup_{b \in C_\ell} R^{-1}(b)$. This is exactly the formula proposed in [107, 108].

Next, we assume $R \subseteq \mathcal{X} \times \mathcal{X}$, now the granules $[a]_R$ are of the form $\rho^{-1}(R(\rho(a)))$.

Corollary 2.5.2. *Let $\lambda : \Omega \to \mathcal{L}$ be a classification and let R be a reflexive relation on $\mathcal{X}$. The lower approximation $\underline{C_\ell}_R$ and upper approximation $\overline{C_\ell}^R$ of C_ℓ w.r.t. R are characterised by*

$$a \in \underline{C_\ell}_R \iff \hat{\lambda}^*_R(a) = \{\ell\} \qquad \textit{and} \qquad a \in \overline{C_\ell}^R \iff \ell \in \hat{\lambda}^*_R(a)\,.$$

Proof.
The composition $\rho^{-1} \circ R \circ \rho$ defines a reflexive relation on Ω with granules of the form $\rho^{-1}(R(\rho(a)))$, whence we can apply the previous proposition. The fact that $\lambda_{\rho^{-1} \circ R \circ \rho}(a) = \hat{\lambda}^*_R(a)$ proves the corollary. □

In this case, we will find $\overline{C_\ell}^R = \bigcup_{b \in C_\ell} \rho^{-1}(R^{-1}(\rho(b)))$.

Variable precision rough sets model. Another popular model is a probabilistic extension of rough sets, called the *variable precision model* [126] (see also Appendix 1.B.2, p. 20). It is worthwhile to notice that the variable precision model was only defined for equivalence relations. The following proposition shows that the variable precision model is based on the distributional interpretation (2.2.3) of the extension of a function $g : X \to Y$ to $g : 2^X \to Y$.

Proposition 2.5.3. *Let $\lambda : \Omega \to \mathcal{L}$ be a classification and let R be a reflexive relation on Ω. Let $\underline{A}_R$ and $\overline{A}^R$ denote the lower and upper approximations w.r.t. R in the variable precision rough sets model of a subset $A \subseteq \Omega$, with $0 \leq \beta < 0.5$. We have for each $\ell \in \mathcal{L}$ that*

$$a \in \underline{C_\ell}_R \iff \lambda_R(a,\ell) \geq 1-\beta \qquad \textit{and} \qquad a \in \overline{C_\ell}^R \iff \lambda_R(a,\ell) > \beta\,.$$

Proof.
For all $a \in \Omega$, and all $\ell \in \mathcal{L}$, it holds that

$$\begin{aligned} \lambda_R(a,\ell) \geq 1-\beta &\iff \lambda(R(a),\ell) \geq 1-\beta \\ &\iff \mathcal{P}(\ell \mid R(a)) = \frac{|\{a \in A \mid \lambda(a) = \ell\}|}{|R(a)|} \geq 1-\beta \\ &\iff \frac{|R(a) \cap C_\ell|}{|R(a)|} \geq 1-\beta\,, \end{aligned}$$

precisely the condition demanded of an object a to belong to the lower approximation of C_ℓ in the variable precision model. □

Of course, we can state a similar proposition in case R is a relation on $\mathcal{X}$.

Decision regions. From the previous propositions, it can be seen that the basic notions of lower and upper approximation from the rough set methodology can be recreated in the framework presented in Sections 2.2 and 2.3. However, these approximations constitute but two different states of a single concept, namely whether the representation $\hat{\lambda}^*_R$ corresponds to a singleton or not. In fact, decision regions play a much more important role than lower and upper approximations since they are (within the set interpretation (2.2.1)) for $\hat{\lambda}^*_R$ what the classes are for λ. Moreover, as Proposition 2.5.1 indicates, the lower and upper approximations can be expressed in terms of decision regions:

$$\underline{C_\ell}_R = \mathbf{C}_\ell \qquad \text{and} \qquad \overline{C_\ell}^R = \bigcup_{L \ni \ell} \mathbf{C}_L\,.$$

Remark that decision regions are more elementary since it is possible to define lower and upper approximation as a union of decision regions, while in the other direction, intersections are needed[10].

[10]This may seem a futile difference, but it is not: in fact, in the article [84] just preceding the article in which the name rough sets was introduced, Pawlak made a point of eliminating the need to perform intersections because of computational expensiveness of this operation.

Also remember that the decision algorithm in rough set data analysis is primarily based on the decision regions, rather than the lower and upper approximations (see Section 1.B.3), even though this is never really stressed in papers/books about rough sets.

Doubt. We can put things in a still slightly different perspective: the (standard) lower approximations can be typified by the non-occurrence of doubt w.r.t. to the labelling, while the upper approximations do allow doubt: i.e. $\underline{C_\ell}$ is the set of objects that are classified as ℓ without any doubt, and $\overline{C_\ell}$ is the set of objects for which there is doubt concerning the exact classification, but we know ℓ is a possibility. This shows that the notion of doubt plays an intrinsic role in rough set methodology.

2.5.3 Model selection

Reducts. The usual approach in defining reducts is to safe-guard the quality of approximation (see Section 1.B.4, p. 21). Such reducts only take into account the lower approximation via the quality of approximation measure $\sum_{\ell \in \mathcal{L}} \frac{|\underline{C_\ell}_R|}{|\Omega|}$. However, this measure fails to grasp some data patterns. Consider for example the information depicted in Table 2.1 and Figure 2.4. While the data is nicely grouped,

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
q	1	1	2	2	3	3	4	4
λ	2	3	2	3	1	4	1	4

Table 2.1: Obvious data pattern, but empty lower approximations.

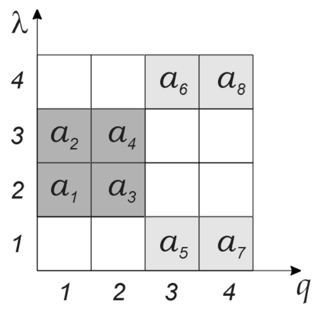


Figure 2.4: Obvious data pattern, but empty lower approximations.

the quality of approximation is zero. The same scheme is imaginable with more attributes, i.e. quality of approximation equal to zero, which would lead to the same 0

for all subsets of these attributes. Any measure based solely on lower approximations will exhibit the same problems.
An obvious alternative would be to demand that a reduct keeps the same lower and upper approximations of the classes C_ℓ. It is easy to see that two relations R_1 and R_2 lead to the same approximations if and only if they lead to the same decision regions, i.e. $\hat{\lambda}^*_{R_1} = \hat{\lambda}^*_{R_2}$ (using the set interpretation (2.2.1)). These decision regions form a partition of Ω, and it is this partition which should remain the same.

A measure for doubt. We can also shift our point of view a bit, and rather stress on the notion of doubt instead of the approximations and decision regions. We can for example measure the amount of doubt present w.r.t. a reflexive relation R. Therefore we define a **doubt relation** D_A on an information granule $A \subseteq \Omega$ between all the objects in A introducing doubt as

DOUBT RELATION.

$$D_A = \{(a,b) \in A^2 \mid \lambda(a) \neq \lambda(b)\}\,,$$

and define the measure

$$\gamma_D(A) = |D_A|\ .$$

First assume we have a partition Π on Ω. We already saw that in this context, doubt may arise within the blocks π, so we can count the (number of couples leading to) doubt in each block:

$$\gamma_D(\pi) = |D_\pi| = \sum_{i \neq j} N_i(\pi) N_j(\pi)\,,$$

where $N_i(\pi) = |\{a \in \pi \mid \lambda(a) = i\}|$. For easier comparison, we should rather create a relative measure, resulting in a figure between 0 and 1. This can, for example, be obtained by dividing the number of couples leading to doubt by the total number of possible couples that can be formed. If $N(\pi) = |\pi|$, we have $N(\pi)^2$ possible couples in the block π and find

$$\begin{aligned}
\frac{\gamma_D(\pi)}{N(\pi)^2} &= \sum_{i \neq j} \frac{N_i(\pi) N_j(\pi)}{N(\pi) N(\pi)} \\
&= \sum_{i \neq j} p(i|\pi) p(j|\pi) \\
&= G(\Lambda|\pi)\,,
\end{aligned}$$

which is exactly the Gini diversity index, one of the most popular measures used in growing decision trees!

2.6 Future research

It would be interesting to extend the whole framework towards graded and fuzzy relations.

APPENDIX

2.A Classifiers and partitions

The following proposition is a very simple one, but it sometimes helps to put things into another perspective. Moreover, it was also the offset for starting to think about what has become Chapter 2.

Proposition 2.A.1. *Consider any classification algorithm that contains no random component. The decision algorithm of the resulting classifier is always partition based.*

Proof.
That the classification algorithm contains no random component just means that its behaviour is totally predictable given the input data. First assume that the algorithm is static, meaning that once the input data has been processed, no further modifications are made to the classifier. As a consequence, the output for any object from Ω is totally determined based upon the initial training parameters (including the training set, and other parameters that have to be fixed in advance). This means that the classifier partitions Ω into different decision regions (output regions), whether this is done explicitly or implicitly.
A non-static algorithm can always be regarded as static at each fixed moment in time t, i.e. we know exactly how it will behave at time t because we know its history until t. Therefore, the decision algorithm at any point in time is partition based, even though the partition may shift in time. $\square$

The previous proposition just says that classification algorithms only work with representations (whether these are flat-line, hierarchical, relational, ...) of objects, not on the objects themselves.

Corollary 2.A.2. *Consider any classifier $\lambda_{cl} : \Omega \to \lambda_{cl}(\Omega)$. At any moment t, there exists a partition Π of Ω such that for all $a \in \Omega$ it holds that $(\forall b \in \Pi(a))(\lambda_{cl}(b) = \lambda_{cl}(a))$.*

Some examples.
NEAREST NEIGHBOURS. Consider the Nearest Neighbour algorithm using Euclidean distance d and assume $\mathcal{X}$ is 2-dimensional. As soon as the classifier is built, all reference points that can act as a nearest neighbour are fixed in $\mathcal{X}$. Now create a Delaunay triangulation[99] of these points in $\mathcal{X}$ and for each triangle T, denote by M_T the center of the circumscribed circle. Now let a, b, c be the vertices of T and let M_{ab}, M_{bc}, M_{ac} be the midpoints of the sides ab, bc, ac. We can now divide each T into three zones Z_a, Z_b, Z_c such that $(\forall \mathbf{x} \in Z_a)(d(\mathbf{x}, a) \leq \min\{d(\mathbf{x}, b), d(\mathbf{x}, c)\})$, see Figure 2.5. These zones obviously form the partition of the decision algorithm. Similar reasonings are valid for higher dimensions and other distances.

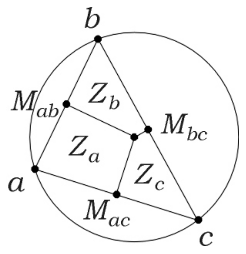


Figure 2.5: Nearest Neighbour partitioning.

RULE BASE. Consider a rule based system. When an object is to be classified, a number of different rules may fire. The outcome of the classification will then depend upon the conflict resolution scheme that is imbedded in the rule based system. But, any time the same vector is prompted to the system, the same rules will fire, and the same conflict resolution system will apply, resulting every time in the same classification (as long as the algorithm is free of random functions).

2.B Decision trees in the classification framework

Introduction. Decision trees are probably about the most characteristic examples of partition-based supervised learning. With their recursive partitioning scheme, they satisfy[11] Lemma 2.3.1 (see p. 36).
Algorithms for the induction of decision trees contain the following more or less interacting ingredients: 1) a splitting rule, 2) a stopping or a pruning rule, and 3) a labelling rule. In this section we will talk about the first and third component.

The splitting rule. There exist a multitude of different splitting rules, but the most popular ones are based on the Shannon entropy and the Gini diversity index.

Extension of the splitting rule towards similarity relations. In Section 2.4.3, we have shown how the Shannon entropy and Gini diversity index can be adapted to deal with similarity relations. It is straightforward to incorporate these changes into splitting rules for tree growing, allowing to soften the usually strict splits. We do not investigate this idea further in this specific context, but we applied this principle in [27] in the context of ranking.
For completeness, we mention the work done in [59]. Work on fuzzy trees (with soft splits) can be found in [3, 21, 100, 111, 119]. Finally, when we talk about including relational information in decision trees, we cannot omit references towards trees in the context of relational databases, e.g. [63, 64, 67, 69].

[11] ID3 is an exception, since it allows "empty" blocks to occur. However, if some block t is empty, the first block t' higher in the recursive scheme that contains t is necessarily non-empty, and ID3 states that t inherits the characteristics of t'.

The labelling rule and decision regions. There exist mainly two types of decision trees: the ones that assign one specific label to each leaf, and the so-called class probability trees that assign a probability distribution to each leaf. A common practice (but not always) to generate the first kind of tree is to start with a class probability tree and just keep the class with highest probability of occurrence in each leaf.
A class probability tree uses the distribution interpretation (2.2.3) of $\hat{\lambda}^*_\Pi$, where Π is of course the partition induced by the tree. This observation implies that we could instead opt for the set interpretation, as is done in the rough set methodology. The tree would then define some decision regions, which in turn can be used to define lower and upper approximations (whether or not this is useful).

Chapter 3

Overview of existing approaches for ranking

Lay-out of this chapter. Here we present a detailed overview and description of the existing techniques that can be found in the literature. However, because of the complexity of the ranking problem, the descriptions of the different methods are very condense, making this chapter maybe a bit more harder to digest in comparison to the others.

This chapter is divided into two main parts. In the first four sections, we discuss specific methods for the problem of learning a complete ranking, where monotonicity plays a crucial role. The last section deals with the problem of ordinal classification (ordinal regression), which has a somewhat longer history.

Notions and conventions. A *partition* (see p. 33) Π of $\mathcal{X}$ is a set of non-empty, pairwise disjoint subsets π of $\mathcal{X}$ such that $\bigcup_{\pi\in\Pi}\pi = \mathcal{X}$. The elements π of a partition Π are called *blocks*. The unique block containing $\mathbf{x} \in \mathcal{X}$ is denoted by $\Pi(\mathbf{x})$.

ORDER (RELATION) $\leq$.

ORDERS AND MONOTONICITY. An **order (relation)** [33] $\leq$ on a set X is a binary relation that is *reflexive* ($a \leq a$), *antisymmetric* (if $a \leq b$ and $b \leq a$, then $a = b$) and *transitive* (if $a \leq b$ and $b \leq c$, then $a \leq c$). As usual, the order $\leq$ decomposes into a *strict order* $<$ and an equality relation $=$. The couple $(X, \leq)$ is called a **poset** (partially ordered set). An order is called **complete** if for all $x_1, x_2 \in X$ either $x_1 \leq x_2$ or $x_2 \leq x_1$.

POSET. COMPLETE.

MONOTONE FUNCTION.

A function $f : (X, \leq_X) \to (Y, \leq_Y)$ between two posets is called **monotone** if for all $x_1, x_2 \in X$ it holds that

$$x_1 \leq_X x_2 \Rightarrow f(x_1) \leq_Y f(x_2)\,.$$

Two elements $x_1, x_2 \in X$ are said to be monotone w.r.t. f if they comply with this rule. By extension, two (input-output) couples $\langle x_1, y_1\rangle, \langle x_2, y_2\rangle \in X \times Y$ are **monotone** w.r.t. each other if the function $f : \{x_1, x_2\} \to \{y_1, y_2\}$ with $f(x_i) := y_i$ is monotone.

MONOTONE (INPUT-OUTPUT) COUPLES.

DATA. The object space Ω is a collection of objects that are described by a finite set of *attributes* $Q = \{q_i : \Omega \to \mathcal{X}_{q_i} \mid i \in N = \{1, \ldots, n\}\}$. Therefore, an object can be represented by a vector $\rho(a) = \mathbf{a} = (q_1(a), \ldots, q_n(a)) \in \mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_{q_i}$.

$\rho, \mathcal{X}_q, \mathcal{X}$

A **learning sample** is a couple $\Lambda = (\mathcal{S}, d)$ where the **sample set** $\mathcal{S}$ is a subset of the object space Ω, and $d : \mathcal{S} \to \mathcal{L}$ a function that assigns a class label to the elements of $\mathcal{S}$. A couple $\langle a, d(a)\rangle$ with $a \in \mathcal{S}$ is called an **example**. In practice, the learning sample Λ is represented by a set of (input-output) couples $\Lambda_{\mathcal{X}} = \{\langle \mathbf{a}, d(a)\rangle \mid a \in \mathcal{S}\}$. We will also use the notation $\mathcal{S}_{\mathcal{X}} = \rho(\mathcal{S}) = \{\mathbf{a} \mid a \in \mathcal{S}\} \subseteq \mathcal{X}$. The learning sample $\Lambda_{\mathcal{X}}$ is called **monotone** if all its elements are monotone w.r.t. each other. Remark that this also implies that if two objects $a, b \in \mathcal{S}$ have the same vector representation $\mathbf{a} = \mathbf{b}$, then it should hold that $d(a) = d(b)$.

LEARNING SAMPLE Λ. SAMPLE SET $\mathcal{S}$. EXAMPLE.

$\Lambda_{\mathcal{X}}, \mathcal{S}_{\mathcal{X}}$

MONOTONE LEARNING SAMPLE.

THE COMPLETE RANKING PROBLEM. In an ordinal classification problem, the aim is to construct a classifier $\lambda : \mathcal{X} \to \mathcal{L}$ based on a learning sample $(\mathcal{S}, d)$, where d is a mapping onto a completely ordered finite set $(\mathcal{L}, \leq_{\mathcal{L}})$, i.e. $d : \mathcal{S} \to (\mathcal{L}, \leq_{\mathcal{L}})$. A detailed discussion about the complete ranking problem can be found in Section 4.2.2 (see p. 89) and Section 4.5 (see p. 97). For the present chapter, it is sufficient to know that the complete ranking problem is in fact a monotone ordinal classification problem, i.e. the classifier λ should be monotone. Also, in a complete ranking problem, we do no longer speak of attributes, but rather of *criteria* (see p. 98), these are functions $c_i : \Omega \to (\mathcal{X}_{c_i}, \leq_{c_i})$. They induce a **product order** $\leq_{\mathcal{X}}$ on $\mathcal{X}$: $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ if and only if $x_i \leq_{c_i} y_i$ for all $i \in N$.

CRITERION.

PRODUCT ORDER $\leq_{\mathcal{X}}$.

STATISTICS. The rank correlation coefficient **Kendall's** τ[1] is defined by

KENDALL'S τ.

$$\tau = \frac{P - Q}{P + Q} \in [-1, 1]\,,$$

where P is the number of concordant pairs[2] $(a, b) \in \mathcal{S}_{\text{test}} \times \mathcal{S}_{\text{test}}$, and Q is the number of discordant pairs (**rank reversal**). This statistic can be interpreted as follows: *"If a pair of objects is sampled at random, the probability that the classifier will rank these objects in the same order is τ higher than the probability that it will rank them in the reverse order."*

RANK REVERSAL.

3.1 Instance-based methods

3.1.1 The approach of Ben-David, OLM

The Ordinal Learning Model was introduced already in 1989 [11], and slightly adapted in 1992 [9].

The general idea. The algorithm consists of a kind of simple conflict resolution scheme w.r.t. non-monotonicity to create a "rule base" B consisting of monotone examples $\langle \mathbf{x}, v \rangle$, with $v \in [1, k] \subset \mathbb{R}$ if $\mathcal{L} = \{1, \ldots, k\}$. Afterwards, OLM assigns a new object a to the highest output value of the vectors smaller than $\rho(a) = \mathbf{a}$, i.e. $\hat{\lambda}_{\text{OLM}}(\mathbf{a}) = \max\{v \in \mathbb{R} \mid \langle \mathbf{x}, v \rangle \in B \wedge \mathbf{x} \leq_{\mathcal{X}} \mathbf{a}\}$.
The construction of the monotone rule base B is essentially based on the following scheme (see [10]): a first example is chosen at random, declared monotone and stored in the rule base B. Afterwards, another example is picked out at random. If this example is monotone w.r.t. the examples already in B, it is also declared monotone and stored in B, otherwise the example is discarded. This procedure

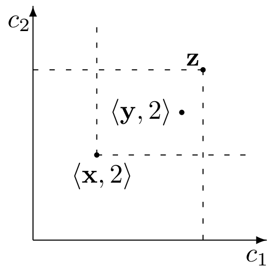
[1]Note that Spearman's rank correlation (treating the ranks as scores and calculating the correlation between the two sets of ranks) is a more widely used measure of rank correlation because of its easier computation. The main advantages of using Kendall's tau are that the distribution of this statistic has slightly better statistical properties and there is a direct interpretation of Kendall's tau in terms of probabilities of observing concordant and discordant pairs. Mostly, these two statistics are very close, and lead to the same conclusions.

[2]A pair (a, b) is called concordant if the order between $d(a)$ and $d(b)$ is maintained between predicted(a) and predicted(b).

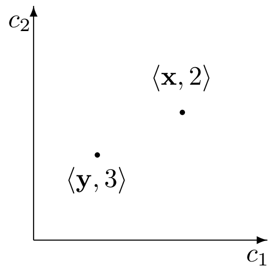
is maintained for all examples in the learning set Λ, resulting in a monotone rule base B.

Building the rule base.

REDUNDANT RULES can occur because of the labelling strategy: if $\langle \mathbf{x}, v\rangle$ belongs to the data base, then any $\langle \mathbf{y}, v\rangle$ with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ does not affect the labelling of new instances $\mathbf{z}$, as can be seen in Figure 3.1(a).


c_2
$\mathbf{z}$
$\langle \mathbf{y}, 2\rangle$
$\langle \mathbf{x}, 2\rangle$
c_1

(a) Building the rule base: redundancy.


c_2
$\langle \mathbf{x}, 2\rangle$
$\langle \mathbf{y}, 3\rangle$
c_1

(b) Building the rule base: non-monotonicity.

Figure 3.1: Redundancy and monotonicity in OLM.

In the somewhat more involved algorithm described in [9], the construction of the rule base B is divided into two phases.

IN THE FIRST PHASE , all doubt is removed from the data by averaging over the classes to which they are assigned, i.e. the new learning set B' is given by the set[3] of $\langle \mathbf{a}, v\rangle$ with $a \in \mathcal{S}$ and where v is the average of the $d(b)$ with $\mathbf{b} = \mathbf{a}$. Remark that taking the mean is not an ordinal operation since it interprets the ordinal scale as an interval scale.

THE SECOND PHASE is initiated with ordering the examples in B' in decreasing output order. Then, the rule base B is constructed by going over the couples of the set B' and by deciding how they influence the rule base B. The first example is added to B. The next example $\langle \mathbf{x}, v\rangle$ is then taken from B'.

(a) If the example is redundant with an example in B, it is simply rejected.

(b) Else, if the example makes an example $\langle \mathbf{y}, w\rangle$ in B redundant, then it is checked whether replacing $\langle \mathbf{y}, w\rangle$ by $\langle \mathbf{x}, v\rangle$ causes conflict in B (B is said to be *conflict-free* if it is monotone and there are no redundant rules). If there are no problems, the replacement is done, otherwise the example $\langle \mathbf{x}, v\rangle$ is rejected.

[3]in the notations of Chapter 2, $B = \{\langle \mathbf{x}, E[\hat{d}(\mathbf{x})]\rangle \mid \mathbf{x} \in \mathcal{S}_{\mathcal{X}}\}$, using the distribution interpretation (2.2.3)

(c) Else, if the example $\langle \mathbf{x}, v\rangle$ is non-monotone w.r.t. a rule $\langle \mathbf{y}, w\rangle$ in B, and $v < w$ (as shown in Figure 3.1(b)), then the same procedure as in (b) is carried out. This step favours lower labels.

(d) Otherwise, add $\langle \mathbf{x}, v\rangle$ to the rule base.

Assignment of labels. As mentioned higher, the labelling of an unseen object $a \in \Omega$ is done by $\hat{\lambda}_{\text{OLM}}(\mathbf{a}) = \max\{v \mid \langle \mathbf{x}, v\rangle \in B \wedge \mathbf{x} \leq_{\mathcal{X}} \mathbf{a}\}$. However, if there are no rules $\langle \mathbf{x}, v\rangle$ in B with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{a}$, then a non-ordinal process is initiated[4]: the nearest rule (in Euclidean distance where $\mathcal{X}_c = \{1, \ldots, k_c\}$ for all criteria c) is fired, or, if this results in more than one rule, the average is taken of the outputs of all these rules. The aim is to prevent very small rule bases from being too conservative.

Open ends.

RANDOMNESS. Of course, this procedure cannot guarantee to produce the same results when repeated on the same data set since it heavily relies upon the order of processing the examples, which happens partially at random: the examples are ordered in decreasing output order, but it is nowhere mentioned how the examples with the same output should be ordered. Consider for example the following subset of examples, $\mathbf{x} = (2,4), \mathbf{y} = (4,5), \mathbf{z} = (1,2)$, all assigned to the same class labelled 2. See Figure 3.2.

- Processing order: $\langle \mathbf{x}, 2\rangle, \langle \mathbf{y}, 2\rangle, \langle \mathbf{z}, 2\rangle$, then $\langle \mathbf{x}, 2\rangle, \langle \mathbf{y}, 2\rangle \in B$.
  Indeed, $\langle \mathbf{z}, 2\rangle$ makes $\langle \mathbf{x}, 2\rangle$ redundant, so it is checked whether replacing $\langle \mathbf{x}, 2\rangle$ by $\langle \mathbf{z}, 2\rangle$ results in a conflict-free data base. This is not the case because $\langle \mathbf{y}, 2\rangle$ is also redundant w.r.t. $\langle \mathbf{z}, 2\rangle$. Therefore, $\langle \mathbf{z}, 2\rangle$ is rejected.
- Processing order: $\langle \mathbf{x}, 2\rangle, \langle \mathbf{z}, 2\rangle, \langle \mathbf{y}, 2\rangle$, then $\langle \mathbf{z}, 2\rangle \in B$.

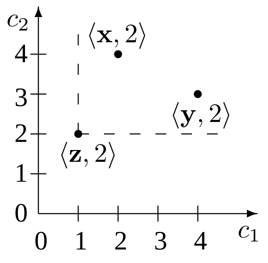
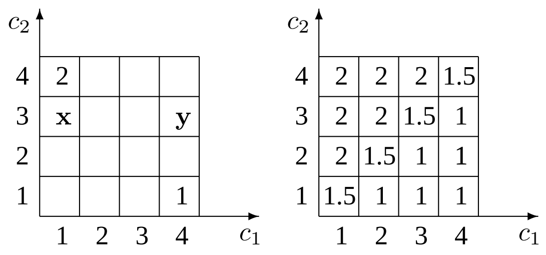

c_2
4
3
2
1
0
$\langle \mathbf{x}, 2\rangle$
$\langle \mathbf{y}, 2\rangle$
$\langle \mathbf{z}, 2\rangle$
0 1 2 3 4
c_1

Figure 3.2: Processing order.

NON-MONOTONE RESULTS are possible following the nearest rule classification scheme. Consider the following rule base $B = \{\langle \mathbf{z}_1, 2\rangle, \langle \mathbf{z}_2, 1\rangle\}$, with $\mathbf{z}_1 = (1,5)$ and $\mathbf{z}_2 = (4,1)$. In Figure 3.3, it is clearly shown that $\mathbf{x} = (1,3)$ will be classified as 2, while $\mathbf{y} = (4,3)$ will be labelled as 1.

[4]The author mentioned that *"this averaging default rule-of-thumb was rarely used [in this experiment]."*



(a) The rule base, and $\mathbf{x}$ and $\mathbf{y}$.

(b) Labelling by OLM.

Figure 3.3: Non-monotone labelling of OLM.

So, while the goal to prevent very small rule bases from being too conservative may be achieved, it seems that the price to pay is the introduction of non-monotonicity in the classifier.
If monotonicity is sacred, then the more conservative approach should be taken: if the rule base B does not contain an example $\langle \min \mathcal{X}, v\rangle$, then add $\langle \min \mathcal{X}, \min \mathcal{L}\rangle$ to B. In that case, the nearest neighbour rule is no longer fired, and the results are guaranteed to be monotone.

Conclusion. Of course, any method can be criticised, and it has to be said that the very simple OLM does deliver very good results in experiments (see Section 5.4 (see p. 130)).

3.2 Decision tree methods

For a short treatise on classification trees, see Appendix 1.A, p. 14.
We start with a global overview of the different methods for learning a ranking by means of decision trees that can be found in the literature. Afterwards, we discuss these methods a bit more in depth.

Notions and conventions.
MONOTONE TREES. Let X_1 and X_2 be two disjoint subsets of a partially ordered set $(X, \leq_X)$, and $y_1, y_2 \in (Y, \leq_Y)$, then it is said that $\langle X_1, y_1\rangle$ and $\langle X_2, y_2\rangle$ are **monotone** w.r.t. each other if the input-output couples $\langle x_1, y_1\rangle$ and $\langle x_2, y_2\rangle$ are monotone w.r.t. each other for all $x_1 \in X_1$ and all $x_2 \in X_2$.
A tree T induces an equivalence relation (a partition) on the data space $\mathcal{X}$, and the leaves t of a tree correspond to the equivalence classes (blocks) denoted by $t_{\mathcal{X}}$.

$t_{\mathcal{X}}$

Therefore, the previous definition of monotone disjoint subsets can be applied to the leaves of a tree. Remark that this definition requires a labelling rule that specifies for each tree how the leaves are to be labelled. A tree is called **monotone** if all its leaves are monotone w.r.t. each other.

MONOTONE LEAVES. MONOTONE TREE.

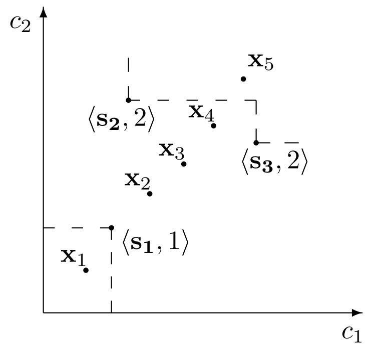
A function $f : (X, \leq_X) \to (Y, \leq_Y)$ is called **quasi-monotone** [74] w.r.t. a subset

QUASI-MONOTONE FUNCTION.

$S \subseteq X$ if for all $x, x' \in X$ it holds that

$$(x \leq_X x') \wedge ([x, x'] \cap S \neq \emptyset) \Rightarrow f(x) \leq_Y f(x')\,.$$

In other words, all elements $x \in X$ must be monotone w.r.t. all elements $s \in S$. As a consequence, we obviously have that $f_{|S}$ must be monotone, and if $s \in S$ is in between two elements $x \leq_X y$ are separated by an (i.e. $x \leq_X s \leq_X y$) then x and y are also monotone w.r.t. each other (because of the transitivity of $\leq_X$ and $\leq_Y$). See also Figure 3.4.


c_2
$\mathbf{x}_5$
$\langle \mathbf{s_2}, 2\rangle$
$\mathbf{x}_4$
$\mathbf{x}_3$
$\langle \mathbf{s_3}, 2\rangle$
$\mathbf{x}_2$
$\langle \mathbf{s_1}, 1\rangle$
$\mathbf{x}_1$
c_1

$$\boxed{\begin{array}{l} f(\mathbf{x}_1) = 1 \leq f(\mathbf{s_1}) \\ f(\mathbf{x}_2) = 2 \in [f(\mathbf{s_1}), f(\mathbf{s_3})] \\ f(\mathbf{x}_3) = 1 \in [f(\mathbf{s_1}), f(\mathbf{s_3})] \\ f(\mathbf{x}_4) = 3 \geq f(\mathbf{s_1}) \\ f(\mathbf{x}_5) = 2 \geq f(\mathbf{s_2}) \end{array}}$$

Figure 3.4: Quasi-monotonicity of f w.r.t. $S = \{\mathbf{s_1}, \mathbf{s_2}, \mathbf{s_3}\}$. Conditions on the sequence $\mathbf{x}_1 \leq_X \mathbf{x}_2 \leq_X \mathbf{x}_3 \leq_X \mathbf{x}_4 \leq_X \mathbf{x}_5$.

3.2.1 Global overview

Methods. The method MID, Monotone Induction of Decision trees [10] was in 1995 the first tree algorithm specifically designed for ranking problems. In 1996, the methods for the binary ranking problem P-DT and QP-DT, Positive Decision Trees and Quasi Positive Decision Trees, saw the light on a symposium, and they were slightly revised in 1999 [74]. Immediately after, in a technical report of 1997, the algorithm MDT, Monotone Decision Trees, was devised as a generalisation of P-DT for the k-class ranking problem [89, 90]. And soon after, also the algorithm QMDT, Quasi Monotone Decision Trees [90, 91], was born as a generalisation of QP-DT.

Characteristics. Classification trees are based on local splitting procedures: the split of each leaf is only based on the data that fall into the leaf. Monotonicity, on the other hand, is a typical global demand. So, to be able to deal with rankings, this global requirement must somehow be incorporated into the local splitting procedure of decision trees. Each of the above mentioned methods has its own way of achieving this: either by altering the *impurity measure* (see p. 16) (as used for classification trees) directly, and/or by affecting it indirectly by adding new data to the training sample during the growing phase.

In Table 3.1, the general characteristics of these methods are put together. In the last column, a reference to the characteristics of standard classification tree algorithms is added.

		RANKING					CLASSIFICATION
		MID	P-DT	QP-DT	MDT	QMDT	C4.5, CART,...
		1995	1996-1999		1997-2002		
WHAT	monotone tree		✓		✓		
	(†) non-monotone data	✓			(∗)		✓
	pruning	✓			(∗)		✓
	visualisation	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	✓
	# classes	≥ 2	2	2	≥ 2	≥ 2	≥ 2
	accuracy on training data	?	100%	100%	100%	100%	?
HOW	alter measure	✓	✓	✓			
	add artificial data		✓	✓	✓	✓	
	parameterised	✓					

(†) non-monotone data also includes stochastic data.

(∗) Very recently, adaptations of this algorithm have been proposed to be able to deal with these problems [17, 18, 87].

Table 3.1: Characteristics (WHAT) of decision tree algorithms for ranking problems, and HOW this is achieved. MID [10], P-DT and QP-DT [74], MDT and QMDT [90].

Overall discussion.
RANKING VERSUS CLASSIFICATION. The ultimate tree algorithm for ranking problems should have the same characteristics as classification trees, and on top of that, be monotone. If we look at Table 3.1, we see that none of the proposed methods satisfies this demand. We can distinguish two approaches:

- MID tries to keep all the characteristics of classification trees while forcing the tree on to a more monotone road by adding a monotone component to the impurity measure used for splitting.

- QP-DT and QMDT focus on the exact monotone ranking of the training data. This is achieved by generating and adding additional artificial data during the growing phase. Like this, the global monotonicity constraint is more or less incorporated into the local environment of the leaves. P-DT and MDT continue this effort, and add artificial data that will finally ensure fully monotone trees.

PRODUCING MONOTONE TREES is not a sinecure: only two of the methods (P-DT and MDT) deliver actual monotone trees. The others have a very good tendency towards monotonicity, measured by some degree of tree monotonicity (proportions of monotone couples of leaves w.r.t. the number of couples of leaves). See however our discussion in Section 5.4.3 (see p. 132) were we argue that the performance on some measure for monotonicity has no direct value (except maybe for the modeler him/herself because (s)he might find a correlation between the degree of monotonicity and the performance on some other measure for the problem at hand).

CONCERNING THE VISUALISATION, it is clear that all the methods can be represented as a tree. However, where this is sufficient to get a comprehensive overview of the rules for classification, this is no longer true for ranking. Indeed, for classification, all rules can be viewed independent from each other, but for rankings the rules must be viewed as a whole because of the monotonicity between them. A tree representation alone is not capable of grasping and bringing out this interrelatedness of the rules (that is, of the leaves). Because their ease of interpretation is one of the most important trumps of decision trees, we sense the lacking of a visualisation of monotonicity as a deficit.

THE SPLIT CRITERION, i.e. the measure for ordering the possible splits, is one of the most essential components in tree growing. For classification trees, a lot of studies on different types of these measure and their properties have been conducted, e.g. [23, 71, 105]. However, the directly adapted or indirectly affected[5] measures used in the present five methods for ranking were never subjected to such a scrutinising study. There is no framework in which they reside, no profound interpretation of these methods is known. And without an interpretation, no in-depth analysis of a ranking problem is possible. (Can we regard the ordering of the possible splits as an ordering of importance[6]? Or does it only put the best split in front, while no conclusions should be drawn from the position of the remaining splits[7]?).

A FEW COMPARISON EXPERIMENTS were made in [10, 74][8]. From these, it became clear that the ranking algorithms outperformed the classification algorithms on accuracy, but in general the resulting trees were more complex (even if the classification trees were not even pruned in these comparisons). Also, there are almost no comparisons on real world domain data, because in practice, data sets are usually not monotone (see also Section 7.1.3 (see p. 184)). Only for MID did we find a report of comparative tests on several real data sets [10], probably because it is the only one capable of handling non-monotone data sets. MID resulted in a lower tree monotonicity degree than C4.5, while maintaining the same performance on the mean square error.

ANOTHER APPROACH , suggested in [91], is to generate a multitude of different classification trees, e.g. via bootstrapping, and check afterwards if they are monotone. A definite plus for this approach is that it can deal with non-monotone data. Of course, it is not guaranteed that a monotone tree will be found.

[5]The addition of data will influence the measure.

[6]If the answer is positive, we could consider the second-best split if for some reason, we would want to disregard the best split. For example in recruitment, it would be too costly to send all applicants to an assessment center, even if this leads to an immediate correct classification.

[7]This question is of particular importance when new data is added to the training set. This new data may be such that a specific kind of split is favoured, but the effect on other splits may be unknown.

[8]Obviously, in [10], the article of 1995 introducing MID, only MID is tested, and in [74], the article on (Q)P-DT, only binary problems are investigated (but all methods are used in the comparison).

3.2.2 The approach of Ben-David, MID

This information-theoretic approach [10] is based on a splitting rule governed by a measure that is the sum of an impurity measure (the *conditional Shannon entropy* (see p. 40)) and a measure of (non-)monotonicity between the leaves.

The measure. At the core of MID, Monotone Induction of Decision trees, lies a new splitting measure, called the total-ambiguity-score, which is constructed as the sum of the conditional Shannon entropy H and a new measure, the order-ambiguity-score, which punishes the degree of non-monotonicity of the tree.
The order-ambiguity-score is a function of the *non-monotonicity index* I which is defined as the ratio of all non-monotonic couples of leaves to the maximum number of all couples of leaves that could be non-monotonic. So, if the tree T has k leaves, we have

$$I(T) = \frac{\sum_{i,j} m_{ij}(T)}{k^2 - k}\,,$$

where $m_{ij}(T) = m_{ji}(T) = 1$ if the couple of leaves (t_i, t_j) is non-monotone, otherwise $m_{ij}(T) = 0$.
Remark that the value of $m_{ij}(T)$, and hence $I(T)$ depends on the labelling rule that is used in the tree T. The *order-ambiguity-score* is now defined as

$$O(T) = \begin{cases} 0 & \text{, if } I(T) = 0 \\ -\frac{1}{\log_2 I(T)} & \text{, if } 0 < I(T) < 1 \\ +\infty & \text{, if } I(T) = 1\,. \end{cases}$$

Finally, the *total-ambiguity-score* is defined as the sum of this score and the conditional Shannon entropy H

$$\Gamma_{\text{MID}}(T) = H(T) + r \cdot O(T)\,,$$

where $r \geq 0$ is a parameter that tries to capture the relative importance of monotonicity relative to the inductive accuracy in a given problem. This measure is to be minimised.

Interpretation. In [10] it is stated that the use of the logarithmic scale in the definition of $O(T)$ is natural because $H(T)$ is logarithmic. However, the measure $O(T)$ has nothing to do with the probabilistic information-theoretic settings of the Shannon entropy (see Section 2.4.2, p. 40). Using the logarithmic scale does not resolve this problem. Therefore, while both measures can be interpreted independently from each other, their addition has no other interpretation besides that it is evidently an aggregation.

Properties. It is clear that non-monotone trees are punished by adding up a score that becomes higher with increasing non-monotonicity. However, the basic assumption for this measure is that impurity and monotonicity are independent concepts, which is not true. These concepts are related in a very subtle, yet definite manner. We will now give some examples of behaviour of the total-ambiguity-score Γ_{MID} that is not (or less) acceptable, and this independent of the choice of the parameter r. We adopt the usual majority rule as labelling rule, but any other rule will lead to similar problems:

a) $\Gamma_{\text{MID}}(00022|111)^9 = \Gamma_{\text{MID}}(000|11122)$, because both trees are monotone according to the majority labelling rule, whence $\Gamma_{\text{MID}}(T) = H(T)$ in both cases,

b) $\Gamma_{\text{MID}}(00022|1111) = 0.133 < \Gamma_{\text{MID}}(000|111122) = 0.148$,

c) $\Gamma_{\text{MID}}(000|111|222) < \Gamma_{\text{MID}}(001|011|222) <$
$< \Gamma_{\text{MID}}(111|000|222) < \Gamma_{\text{MID}}(011|001|222)$.

Note that all these examples only look at the simplest case of the split of the root node, where the leaves are ordered according to a simple linear order. These examples show that the measure may not behave as would be expected. For a) we would expect $\Gamma(00022|111) > \Gamma(000|11122)$, for b) $\Gamma(00022|1111) > \Gamma(000|111122)$, and for c) we would expect $\Gamma(000|111|222) < \Gamma(001|011|222) < \Gamma(011|001|222) < \Gamma(111|000|222)$. (In [27] we probed a bit deeper into the expected behaviour of measures for a ranking.)

Assignment of labels. As is usual in decision trees.

Input-output. A good property of MID is that it is capable of dealing with non-monotone data sets, which is a big advantage in practice. It also makes sure that all available data is taken into account, i.e. one does not need to eliminate some examples from the data set $\Lambda = (\mathcal{S}, d)$ in order to render $\Lambda_{\mathcal{X}} = \{\langle \mathbf{a}, d(a)\rangle \mid a \in \mathcal{S}\}$ monotone. This also leads to more flexibility, since it is possible to run MID using any subset of criteria one might be interested in for further investigation (even if this subset does not guarantee the monotonicity of the learning sample). However, the resulting tree is not guaranteed to be monotone, not even when the learning sample is monotone.

Some remarks. In Chapter 7, Section 7.4.1 (see p. 200), we will show how we can find a monotone labelling for any tree, therefore also for MID. Nevertheless, it should be noted that this mending strategy does not guarantee good results because the underlying philosophy behind it is different from the one underlying MID (the latter regards impurity and non-monotonicity as independent, and assumes that there is a trade-off possible between these two concepts, while the former is based on the opposite idea, focussing on the interaction between impurity and non-monotonicity.)

[9] This should be read as: a split of the root node in two children t_L and t_R, where t_L contains 3 class-0 objects, and 2 class-2 objects, and t_R contains 3 class-1 objects. Similarly, (000|111|222) denotes a split of the root node into 3 children.

3.2.3 The approaches of Makino et al., (Q)P-DT, and of Potharst and Bioch, (Q)MDT

In Japan, Makino et al. [74] looked deeper into the most basic situation of the ranking problem: build binary decision trees that represent (quasi-)monotone functions that are discriminating for the learning sample $\mathcal{S}$ (i.e. they correctly reclassify all objects in $\mathcal{S}$) when there are only two classes. In the two-class problem, monotone functions are also called positive functions, hence the name (Quasi-)Positive Decision Tree, or short (Q)P-DT, for their method.

A bit later, on the other side of the world in the Netherlands, these methods were (non-trivially) adapted to handle n-ary trees for the k-class problem by Potharst and Bioch [88, 89, 90, 91]. They baptised their method (Q)MDT, short for (Quasi-)Monotone Trees. Both approaches, (Q)P-DT and (Q)MDT, are based on known impurity measures, but rather than significantly altering the measure itself to incorporate monotonicity, they use *updating rules* during the growing phase to ensure the (quasi-)monotonicity of the final tree. These updating rules alter the original learning sample by adding to it specific objects with well-chosen labels.

The measure. The most obvious difference between both approaches concerns the impurity measure that is used. (Q)P-DT uses a slightly modified version of the (binary) Shannon entropy, namely

$$H^{+}(p_1, p_2) = \begin{cases} 1 & \text{, if } p_1 < p_2 \\ H(p_1, p_2) & \text{, otherwise}, \end{cases} \tag{3.2.1}$$

and, if t_L and t_R denote the two children of a node t after a split $c_i \leq_{c_i} v$, the split of t (among the splits with monotone children[10] if these exist, among all possible splits otherwise) that minimises

$$\Gamma_{\text{PDT}}(t_L, t_R) = \hat{p}\,(t_L) \cdot H^{+}(\hat{p}\,(1|t_L), \hat{p}\,(2|t_L)) + \hat{p}\,(t_R) \cdot H^{+}(\hat{p}\,(2|t_R), \hat{p}\,(1|t_R))$$

is taken. They also tested some other measures I instead of H^{+}, all defined according to $I(p, 1-p) := I'(p)$ where I' is non-increasing and satisfies $I'(0.5) = 1$ and $I'(1) = 0$.

(Q)MDT, on the other hand, sticks to the tree impurity measures known from classification problems and minimises directly among all splits. It should be remarked that, indeed, it would be an extremely difficult task to generalise (3.2.1) towards k classes in a meaningful and useful manner.

Remark that these measures are still only locally defined, and that the main engine driving towards monotonicity consists of the updating rules that gradually add data to the learning sample:

[10] If $(\mathcal{L}, \leq_{\mathcal{L}}) = \{1 \leq 2\}$, they define two children as monotone if either t_L does not contain class-2 examples, or if t_R does not contain class-1 examples.

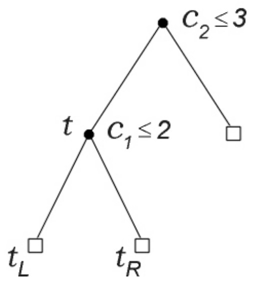
Cornering and bordering: the updating rules. At the core of these methods are the updating rules, that enter new examples into the learning sample. There are two different techniques, one is called *cornering* in [89], the other one could be called *bordering*. They can be executed separately or together as shown in Table 3.2. From this table, it can be seen that bordering leads to quasi-monotonicity, and cornering to monotonicity.

	QP-DT	P-DT	QMDT	MDT
bordering	✓	✓	✓	
cornering		✓		✓

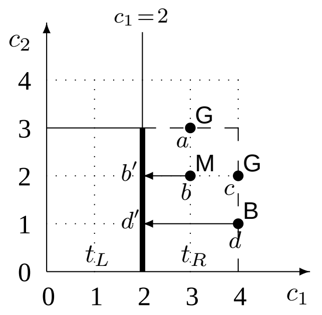
Table 3.2: Cornering and bordering.

BORDERING. Let $\Lambda_{\mathcal{X}}(t)$ be the subset of learning examples that fall into the leaf t. Now perform a univariate binary split $c_i \leq_{c_i} v$ on t, resulting in the children t_L and t_R. Instead of simply continuing splitting using the sets of examples $\Lambda_{\mathcal{X}}(t_L)$ and $\Lambda_{\mathcal{X}}(t_R)$, some additional examples are added to these sets: assuming $\mathcal{L} = \{1, \ldots, k\}$

- examples to add to $\Lambda_{\mathcal{X}}(t_L)$: the projection of "current data in t_R with associated label $<_{\mathcal{L}} k$" along "the axis c_i" onto "the border of the block $(t_L)_{\mathcal{X}} \subset \mathcal{X}$ determined by the plane $c_i = v$" (see Figure 3.5). These vectors $\mathbf{x}$ are labelled with the lowest value $\lambda_{\min}(\mathbf{x})$ that still ensures the monotonicity of the new data set, i.e. $\lambda_{\min}(\mathbf{x}) = \max\{d(a) \mid a \in \mathcal{S} \wedge \mathbf{a} \leq_{\mathcal{X}} \mathbf{x}\}$, or 1 if the set $\{a \in \mathcal{S} \wedge \mathbf{a} \leq_{\mathcal{X}} \mathbf{x}\}$ is empty. $\lambda_{\min}$



(a) Tree.



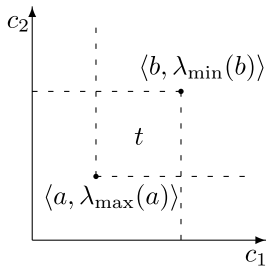
(b) Bordering for t_L when the leaf t_R initially contains the 4 examples $\langle a, \mathsf{G}\rangle, \langle b, \mathsf{M}\rangle, \langle c, \mathsf{G}\rangle, \langle d, \mathsf{B}\rangle$ after the splitting of t.

Figure 3.5: Bordering for the left child t_L. $\mathcal{L} = \{$B(ad) $<_{\mathcal{L}}$ M(oderate) $<_{\mathcal{L}}$ G(ood)$\}$.

- examples to add to $\Lambda_{\mathcal{X}}(t_R)$: the projection of "current data in t_L with associated label $>_{\mathcal{L}} 1$" along "the axis c_i" onto "the operational border of t_R, i.e. the intersection of $(t_R)_{\mathcal{X}}$ with the plane $c_i = v^+$", where $v^+ := \min\{w \in \mathcal{X}_{c_i} \mid (\exists \mathbf{a} \in \mathcal{S}_{\mathcal{X}})(a_i = w) \wedge (w >_{c_i} v)\}$. These vectors $\mathbf{x}$ are labelled with the highest value $\lambda_{\max}(\mathbf{x})$ that still ensures monotonicity of the new data set, i.e. $\lambda_{\max}(\mathbf{x}) = \min\{d(a) \mid a \in \mathcal{S} \wedge \mathbf{a} \geq_{\mathcal{X}} \mathbf{x}\}$, or k if the set $\{a \in \mathcal{S} \wedge \mathbf{a} \geq_{\mathcal{X}} \mathbf{x}\}$ is empty. $\lambda_{\max}$

Figure 3.5 makes it clear why the bordering method leads to quasi-monotone trees: if splitting is continued until all examples in the leafs belong to the same class, then these projections guarantee the monotonicity w.r.t. to samples $\mathcal{S}_{\mathcal{X}}$.

CORNERING. To create fully monotone trees, another approach is needed. Assume that $\mathcal{X}$ is finite (the continuous case is considered in the footnotes). In [90] it was proven that every block $t_{\mathcal{X}}$ corresponds to an interval[11] $[\min t_{\mathcal{X}}, \max t_{\mathcal{X}}] = [a(t), b(t)] \subseteq \mathcal{X}$.
Now, these corner vectors[12] of the blocks $t_{\mathcal{X}}$ are added to the samples $\mathcal{S}_{\mathcal{X}}(t)$ falling into t because these vectors affect all vectors inside $t_{\mathcal{X}}$. The minimal vector $a(t)$ is labelled with the maximal label $\lambda_{\max}(a(t))$ that ensures monotonicity in the new data set, and the maximal vector $b(t)$ is labelled likewise with the minimal label $\lambda_{\min}(b(t))$ (see Figure 3.6). Like this, the range of possible labels within a node is kept to its strict minimum while assuring monotonicity.


c_2
$\langle b, \lambda_{\min}(b)\rangle$
t
$\langle a, \lambda_{\max}(a)\rangle$
c_1
Figure 3.6: Cornering of a leaf t.

REMARK. Although some techniques for adding generated data have already proven useful for classification purposes, e.g. [22], it is not clear what are the side effects – besides monotonicity – of the updating rules in the framework of ranking. One such an effect has been described in [18, 87] when applied to non-monotone data (see also the remark at the end of this section): they remark that the updated data set grows exponentially, a property directly linked to the tree size.

Properties. In the test setting of [74], P-DT and QP-DT delivered in general better results than MID on monotone learning samples (but do not forget that MID

[11] In the continuous case, $t_{\mathcal{X}} =]a(t), b(t)]$.
[12] In the continuous case, the operational minimal vector $a'(t)$ must be taken instead of $a(t)$.

can also deal with non-monotone data sets). It was however remarked that MID produced smaller trees.
In general, we found that the updating rules (both cornering and bordering) may sometimes lead to more or less *blind splits*, i.e. the choice must be made between a number of splits that lead to the same value on the splitting measure (see Appendix 3.A (see p. 80)). In our experiments, we found that this could cause MDT, which only uses cornering, to produce extremely large trees even for small training samples (experimental results can be found in Appendix 3.B (see p. 80)). It has not yet been tested if and how much of this is alleviated by adding the bordering procedure.

Interpretation. Although the methods deliver good results, so far, there is no interpretation of the impurity measures in combination with an updating rule.

Assignment of labels. The algorithm only stops when all leafs are pure (after updating the samples). This makes the assignment of labels to the leafs self-evident.

Input-output. These methods demand a monotone data set to start with. The final tree is either monotone, or quasi-monotone, depending on the algorithm used. As for MID, the relabelling technique of Section 7.4.1 (see p. 200) can be used to transform a quasi-monotone tree into a monotone one.

Some remarks.
NON-MONOTONE DATA. Parallel with our own research on monotone learning algorithms (described in the next four chapters), Popova and Bioch revised the cornering technique to make the algorithm MDT capable of processing non-monotone data. In fact, the algorithm stays exactly as described above, but whereas in the monotone case, the original training data is left unchanged, this is no longer true in the non-monotone case: the corner vectors $a(t)$ and $b(t)$ are always labelled with resp. $\lambda_{\max}(a(t))$ and $\lambda_{\min}(b(t))$, even if this implies reassigning the labels of existing examples.
LABELLING AND PRUNING. At the same time, they proposed a pre- and post-pruning technique for monotone trees based on monotone labelling techniques. We come back to these in Section 7.4 (see p. 200).

3.3 Rough set methods

For a short treatise on rough sets, see Appendix 1.B, p. 17.

Notions and conventions. Let $Q = \{q_i : \Omega \to \mathcal{X}_i \mid i \in N = \{1, \ldots, n\}\}$ be a finite set of attributes, and let $I \subseteq N$. Denote $\mathcal{X}_I = \prod_{i \in I} \mathcal{X}_i$. $\mathcal{X}_I$
Based on the granules $[\mathbf{a}]_{\mathcal{X}_I} = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{b} =_{\mathcal{X}_I} \mathbf{a}\}$, the *upper* and *lower approximations* (see p. 19) of a subset $A \subseteq \mathcal{S}$ w.r.t. I are denoted by $\overline{A}^I = \bigcup_{a \in A} \rho_{|\mathcal{S}}^{-1}([\mathbf{a}]_{\mathcal{X}_I})$, $\overline{A}^I, \underline{A}_I$

and $\underline{A}_I = \bigcup_{a\in A} \rho_{|\mathcal{S}}^{-1}(\{[\mathbf{a}]_{\mathcal{X}_I} \mid [\mathbf{a}]_{\mathcal{X}_I} \subseteq \rho(A)\})$, respectively grouping the objects that possibly or certainly belong to A. The associated *boundary region* is denoted by $\delta_I(A) = \overline{A}^I \setminus \underline{A}_I$.

$\delta_I(A)$

C_ℓ (altered definition of C_ℓ for this section!)

In the supervised learning context, the classes[13] $C_\ell = \{a \in \mathcal{S} \mid d(a) = \ell\}$ are approximated. A sample object $a \in \mathcal{S}$ is said to be *consistent* w.r.t. I if does not belong to one of the boundary regions $\delta_I(C_\ell)$, these objects can be correctly reclassified based on the set of attributes $\{q_i \mid i \in I\}$. The *quality of approximation* γ_I (see p. 21) is then defined as the ratio of consistent sample objects w.r.t. all sample objects, or equivalently, $\gamma_I = |\underline{C_\ell}_I|/|\mathcal{S}|$.

General remark. There is no literature in which the following rough set adaptations are extensively compared with each other or with any other method. Only in [48], DBRS (Dominance-based Rough Set approach) and OO (Ordinal-Ordinal) are compared on one very small data set[14]: 15 objects and 4 criteria.

3.3.1 The approach of Greco et al., DBRS

From 1995 on, Greco et al. [50, 51, 52, 53] pursued the adaption of the rough set approach towards ranking problems. They ground their philosophy in multi-criteria decision aid (MCDA) [97] and preference modelling in order to deal with examples that are inconsistent with the monotonicity requirement.
We will eliminate references towards MCDA and preference modelling in our overview of their method, and tell the story more in line with the general ideas of Chapter 2. However, Greco et al. should get credit for the idea of blending these fields with rough sets methodology for supervised learning. In fact, the idea of incorporating the philosophy of MCDA within machine learning formed the seedling for all our work on the supervised learning of a ranking.

The approximations. Instead of approximating the classes $C_\ell = \{a \in \mathcal{S} \mid d(a) = \ell\}$ based on the partial equality relation $=_{\mathcal{X}_I}$ (i.e. based on the granules $[\mathbf{a}]_{\mathcal{X}_I} \subseteq \mathcal{X}$), they consider the approximation of cumulative classes $C_{\geq\ell} := \bigcup_{i\geq_{\mathcal{L}}\ell} C_i$ and $C_{\leq\ell} := \bigcup_{i\leq_{\mathcal{L}}\ell} C_i$ based on the partial order relation $\geq_{\mathcal{X}_I}$ with granules of the form $[\mathbf{a})_{\mathcal{X}_I} = \{\mathbf{b} \in \mathcal{X} \mid \mathbf{b} \geq_{\mathcal{X}_I} \mathbf{a}\}$, and $(\mathbf{a}]_{\mathcal{X}_I} = \{\mathbf{b} \in \mathcal{X} \mid \mathbf{b} \leq_{\mathcal{X}_I} \mathbf{a}\}$. They define (see Figure 3.7)

$$\overline{C_{\geq\ell}}^I = \bigcup_{a\in C_{\geq\ell}} \rho_{|\mathcal{S}}^{-1}\left([\mathbf{a})_{\mathcal{X}_I}\right),$$
$$\underline{C_{\geq\ell}}_I = \bigcup_{a\in C_{\geq\ell}} \rho_{|\mathcal{S}}^{-1}\left(\left\{[\mathbf{a})_{\mathcal{X}_I} \mid [\mathbf{a})_{\mathcal{X}_I} \subseteq \rho(C_{\geq\ell})\right\}\right),$$

[13]In order to avoid notational conflicts, we should write $C_\ell \cap \mathcal{S}$, because in other chapters, we have maintained the notation $C_\ell = \{a \in \Omega \mid d(a) = \ell\}$. However, this would burden the notation too much, so we ask the reader to keep in mind this altered notation in this section.
[14]The contraception data set from [31], see also Table 7.3.3, p. 197.

and likewise,

$$\overline{C_{\leq \ell}}^{I} = \bigcup_{a \in C_{\leq \ell}} \rho^{-1}_{|\mathcal{S}} \Big((\mathbf{a}]_{\mathcal{X}_I} \Big) ,$$

$$\underline{C_{\leq \ell}}_{I} = \bigcup_{a \in C_{\leq \ell}} \rho^{-1}_{|\mathcal{S}} \Big(\Big\{ (\mathbf{a}]_{\mathcal{X}_I} \mid [\mathbf{a})_{\mathcal{X}_I} \subseteq \rho(C_{\leq \ell}) \Big\} \Big) .$$

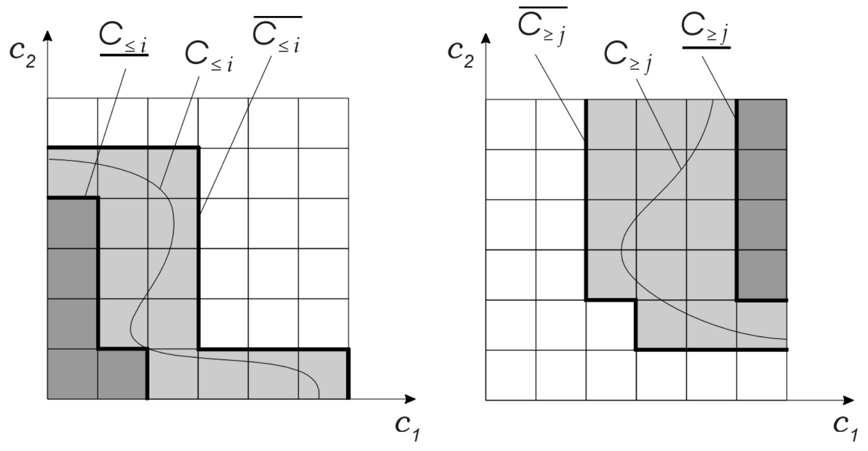


Figure 3.7: Upper and lower approximations in DBRS.

A sample object $a \in \mathcal{S}$ is called *consistent* if it does not belong to one of the boundary regions, i.e. if

$$a \notin \delta_I(\mathcal{S}, d) := \bigcup_{\ell \in \mathcal{L}} \delta_I(C_{\geq \ell}) \cup \bigcup_{\ell \in \mathcal{L}} \delta_I(C_{\leq \ell}) .$$

From these approximations, they then derive a *quality of sorting* γ_I^{DBRS} as the ratio of consistent sample objects w.r.t. all sample objects, i.e.

$$\gamma_I^{\text{DBRS}}(\mathcal{S}, d) := \frac{|\mathcal{S} \setminus \delta_I(\mathcal{S}, d)|}{|\mathcal{S}|} .$$

The rules. Based on this quality of sorting, reducts can be defined, and from these reducts, rules can be generated as described in [52]. They resort to five type of rules, with the following three basic types (let us call them **inequality rules**): INEQUALITY RULES.

(i) if $c_{i_1}(a) \leq_{c_{i_1}} v_1$ and ... and $c_{i_s}(a) \leq_{c_{i_s}} v_s$, then $\lambda'(a) \leq_{\mathcal{L}} \ell$,

(ii) if $c_{i_1}(a) \geq_{c_{i_1}} v_1$ and ... and $c_{i_s}(a) \geq_{c_{i_s}} v_s$, then $\lambda'(a) \geq_{\mathcal{L}} \ell$,

(iii) if $c_{i_1}(a) \leq_{c_{i_1}} v_1$ and ... and $c_{i_s}(a) \leq_{c_{i_s}} v_s$ and
$c_{i_1}(a) \geq_{c_{j_1}} w_1$ and ... and $c_{i_s}(a) \geq_{c_{j_t}} w_t$, then $\lambda'(a) \in [\ell_1, \ell_2]$,

where $[\ell_1, \ell_2]$ is an interval in $(\mathcal{L}, \leq_{\mathcal{L}})$. Depending on which kind of approximation of $C_{\leq \ell}$ is used, the first two types are divided in *certain* (lower approximation) and *possible* (upper approximation) rules.
Only one basic type of inequality rules may be used at a time in a rule base to make sure there are no monotonicity problems. However, when generating *minimal*[15] sets of rules, the authors generate rule sets comprising all these types of rules together (at least in their examples).

Assignment of labels. When only one type of rule is applied, the following scheme is suggested in [51]: if an object $a \in \Omega$ matches several rules of

- type (i): then a is assigned to the minimum of the corresponding labels.
- type (ii): then a is assigned to the maximum of the corresponding labels.
- type (iii): then a is assigned to the union of the intervals.

Open ends. Even though they consider rule sets gathering rules of different types, it is not discussed how possible conflicts between them can be resolved. For example, what happens if an object a is classified as $\lambda_{\text{DBRS}}(a) \leq_{\mathcal{L}} 2$ by one rule, and as $\lambda_{\text{DBRS}}(a) \geq_{\mathcal{L}} 4$ by another?
Moreover, they do neither prove nor contradict the monotonicity of such a mixed rule base. Purely based on the different types of rules, it is possible to define a non-monotone mixed rule base:

- if $c_1(x) \leq_{c_1} 2$, then $\lambda'(x) \leq_{\mathcal{L}} 4$,
- if $c_2(x) \geq_{c_2} 3$, then $\lambda'(x) \leq_{\mathcal{L}} 2$.

If we apply the assignment rules mentioned above, then the objects a, b, with $c_1(a) = 1, c_2(a) = 2$ and $c_1(b) = 3, c_2(b) = 3$ lead to $\lambda'(a) = 4 \geq_{\mathcal{L}} \lambda'(b) = 2$, while $\mathbf{a} \leq_{\mathcal{X}} \mathbf{b}$. It should be shown whether the algorithms exclude this kind of configurations, and if not, it would be interesting to look for different algorithms and/or conflict resolution schemes to remedy this.

Remarks.
AN ALTERNATIVE TO THE APPROXIMATIONS. The framework we will develop in Chapter 4 leads to an alternative that is closer to the original rough set description: replace the relational representation in Corollary 2.5.2 by the consistent interval representation (based on the set interpretation $(2.2.1)$) as defined in Theorem 4.6.3. This leads to

$$a \in \underline{C_\ell}_I \iff \tilde{d}^*_{\Pi_I}(a) = \{\ell\} \qquad \text{and} \qquad a \in \overline{C_\ell}^I \iff \ell \in \tilde{d}^*_{\Pi_I}(a)\,.$$

The resulting ratio between consistent sample objects (i.e. objects $a \in \mathcal{S}$ with $|\tilde{d}^*_{\Pi_I}(a)| = 1$) and all sample objects then just coincides with γ_I^{DBRS}.

[15] A set of rules is minimal if it reclassifies all consistent examples correctly and inconsistent objects are classified to clusters of classes referring to this inconsistency. Moreover, removing one rule makes the previous condition false.

CONCERNING THE QUALITY OF SORTING. Gediga and Düntsch in [48] formulated a serious deficit in the combined action of the quality of sorting γ_I^{DBRS} and the three basic types of inequality rules that are sought after. They give as an example Table 3.3.

(a) Learning sample.

	a_1	a_2	a_3	a_4	a_5	a_6
c	2	1	4	3	6	5
d	1	2	3	4	5	6

(b) The mapping d has an obvious pattern, but all objects are inconsistent.

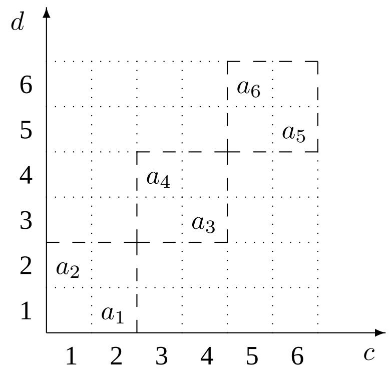


Table 3.3: Disassociation of γ_I^{DBRS} and the inequality rules.

We find that $\gamma_I^{\text{DBRS}}(\mathcal{S}, d) = 0$, while there are clearly several rules valid on this learning sample:

$$\begin{aligned} c(x) \leq_c 2 \Rightarrow d(x) \leq_{\mathcal{L}} 2\,, &\quad c(x) \leq_c 4 \Rightarrow d(x) \leq_{\mathcal{L}} 4\,, \\ c(x) \geq_c 3 \Rightarrow d(x) \geq_{\mathcal{L}} 3\,, &\quad c(x) \geq_c 5 \Rightarrow d(x) \geq_{\mathcal{L}} 5\,. \end{aligned}$$

In the next section, we summarise the alternative proposed in [48]. But let us first have a closer look at this remark.

The problem lies in the wrong extension of the concept *inconsistency* w.r.t. inequality rules. In fact, Greco et al. have rather extended the concept of inconsistency w.r.t. equality rules (if $c_{i_1}(a) = v_1$ and ... and $c_{i_s}(a) = v_s$, then $\lambda'(a) = \ell$), a statement that is supported by the previous remark about the alternative definitions of upper and lower approximations. Seen from this angle, this example just lays bare a default of the original definition of quality of approximation, as already pointed out in Section 2.5.3, with the example in Table 2.1 and Figure 2.4 (see p. 46). Nevertheless, it also shows that *certain* (as opposed to *possible*) equality rules are harder to satisfy for ranking problems than for classification problems, and this seems to indicate that for the ranking problem, focussing on this kind of certain equality rules is probably not a good path to follow, at least not when the learning sample is allowed to be non-monotone.

3.3.2 The approach of Gediga and Düntsch, OO

The method OO, Ordinal-Ordinal [48][16], was a reaction of Gediga and Düntsch on the previous method. More precisely, it proposes another measure: *quality of sorting rules*, stressing the relation between the measure and the inequality rules sought after. However, these rules are a bit more restricted in that the left-hand side of one rule base is always based on the evaluations of all criteria c_i within a fixed set $I \subseteq N$. In terms of the classification problem, this means that each block in $\mathcal{X}$ induced by $\mathcal{X}_I$ represents a rule, so there is no second step of rule generation involved anymore. In other words, the rules are of the form *"if* $c_1(a) = v_{c_1}$ *and ... and* $c_k(a) = v_{c_k}$*, then* $\lambda'(a) = \ell$*"*, or equivalently, *"if* $\mathbf{a} \in [\mathbf{x}]_{\mathcal{X}_I}$*, then* $\lambda'(a) = \ell$*"*, with $\mathbf{x} \in \mathcal{X}$.
Beside introducing a new measure, they also proposed some statistical techniques, within the framework for rough sets they developed in [46, 47], to analyse the significance of the obtained reducts. This is more or less in the same spirit as the idea of dynamic reducts [7], except that some parameters need to be set to calculate dynamic reducts.

The measure.

CLASSIFICATION. If in the usual rough set approach, equality rules of the form "if $c_{i_1}(a) = v_1$ and ... and $c_{i_s}(a) = v_s$, then $\lambda'(a) = \ell$" are used, as discussed above, then it can be observed that the standard quality of approximation can be interpreted as a two-step aggregation of such rules:

(i) for fixed $\ell \in \mathcal{L}$, an aggregation of the covering degree of all certain rules of type *"if* $c_{i_1}(a) = v_1$ *and ... and* $c_{i_s}(a) = v_s$*, then* $\lambda'(a) = \ell$*"*, leading to a quality of *"then* $\lambda'(a) = \ell$*"*-rules.

(ii) an aggregation over all $\ell \in \mathcal{L}$ of the quality of *"then* $\lambda'(a) = \ell$*"*-rules.

The covering degree of a rule *"if left-hand side is true, then right-hand side is true"* is simply the proportion of the number of sample objects that satisfy both left-hand side and right-hand side to the number of sample objects that satisfy the right-hand side. In the particular setting of OO, this means

$$\gamma'_I(\mathbf{a} \in [\mathbf{x}]_{\mathcal{X}_I} \Rightarrow \lambda'(a) = i) := \frac{|\{a \in C_i \mid \mathbf{a} \in [\mathbf{x}]_{\mathcal{X}_I}\}|}{|C_i|}\,.$$

The aggregation (i) of these qualities is a simple addition, but only of the *certain* rules, i.e. all objects that satisfy the left-hand side must also satisfy the right-hand side. Hence we obtain for fixed $\ell \in \mathcal{L}$

$$\gamma'_I(C_i) \;:=\; \sum_{[\mathbf{x}]_{\mathcal{X}_I} : [\mathbf{x}]_{\mathcal{X}_I} \subseteq \rho(C_i)} \gamma'(\mathbf{a} \in [\mathbf{x}]_{\mathcal{X}_I} \Rightarrow \lambda'(a) = i)\,.$$

[16]The method OO is in fact part of NOO, Nominal-Ordinal-Ordinal, that also allows nominal attributes.

The second aggregation (ii) is a weighted sum, where the weights are the weights of the classes C_ℓ w.r.t. all classes. It is not that hard to see that the final result just corresponds to the standard quality of approximation

$$\gamma_I(\mathcal{S}, d) = \sum_{i\in\mathcal{L}} \frac{|C_i|}{\sum_{j\in\mathcal{L}} |C_j|} \times \gamma'_I(C_i) = \sum_{i\in\mathcal{L}} \frac{|\underline{C_i}_I|}{|\mathcal{S}|} .$$

RANKING. The idea is now to apply the previous to inequality rules of type (i), $\leq\Rightarrow\leq$, and of type (ii), $\geq\Rightarrow\geq$, using the cumulative classes $C_{\leq\ell}$ and $C_{\geq\ell}$. Denoting $\mathcal{L} = \{1, \ldots, k\}$, this results in

$$\gamma'_I(\mathbf{a} \in [\mathbf{x})_{\mathcal{X}_I} \Rightarrow \lambda'(a) \geq_{\mathcal{L}} i) := \frac{|\{a \in C_{\geq i} \mid \mathbf{a} \in [\mathbf{x})_{\mathcal{X}_I}\}|}{|C_{\geq i}|} ,$$

$$\gamma'_I(C_{\geq i}) := \sum_{[\mathbf{x})_{\mathcal{X}_I} : [\mathbf{x})_{\mathcal{X}_I} \subseteq \rho(C_{\geq i})} \gamma'(\mathbf{a} \in [\mathbf{x})_{\mathcal{X}_I} \Rightarrow \lambda'(a) \geq i) ,$$

$$\gamma_I^{(\geq,\geq)}(\mathcal{S}, d) := \sum_{i=2}^{k} \frac{|C_{\geq i}|}{\sum_{j=2}^{k} |C_{\geq j}|} \times \gamma'_I(C_{\geq i}) ,$$

where $i = 1$ is not included in the summation because satisfying the right-hand side “then $\lambda(a) \geq 1$” does not add any knowledge.
Likewise, $\gamma_I^{(\leq,\leq)}$ can be defined. If we now write $\gamma_I^{(\geq,\geq)}(\mathcal{S}, d) = A/B$ and $\gamma_I^{(\leq,\leq)}(\mathcal{S}, d) = C/D$, then the measure γ_I^{oo} is finally defined as a mean of the previous two measures:

$$\gamma_I^{\mathrm{oo}} := \frac{A + C}{B + D} .$$

Statistical issues. Based on random permutations and bootstrap simulations, they devise a way to gauge the “real prediction effect”. They also demonstrate that $\frac{1+\tau}{2}$, where τ is Kendall’s tau, forms an upper bound of both $\gamma_I^{(\leq,\leq)}$ and $\gamma_I^{(\geq,\geq)}$.

3.3.3 The approach of Bioch and Popova, MRSA

A radically different approach, MRSA (Monotone Rough Set Approach), is advocated by Bioch and Popova in [15, 16, 87]. Their theory focusses on boolean reasoning based on the use of the discernibility matrices and discernibility functions in rough set theory. For classification, the approach based on the quality of approximation and the approach based on discernibility matrices and functions coincide.

Monotone reducts. Starting from a monotone learning sample $(\mathcal{S}, d)$ with $\mathcal{S} = \{a_1, \ldots, a_m\}$, and a set of criteria C, the *monotone discernibility matrix* $M(\mathcal{S}, d)$ is defined as

$$M_{ij} = \begin{cases} \{c \in C \mid c(a_i) >_c c(a_j)\} & \text{for } i, j : d(a_i) >_{\mathcal{L}} d(a_j) \\ \emptyset & \text{otherwise.} \end{cases}$$

In the same manner the discernibility function is derived from the discernibility matrix [65], the *monotone discernibility function* can be derived from the monotone discernibility matrix. The *monotone reducts* are then the minimal transversals of the entries of the monotone discernibility matrix. (For more details and explanations, see [15, 16, 87]).

Rule generation. They propose two alternatives: producing monotone minimal covers for a certain monotone reduct, leading to certain inequality rules of type (i) and (ii), or producing a set of monotone rules (with equality for the class value) describing the maximal extension $\lambda_{\max}$ of the learning sample $(\mathcal{S}_{\mathcal{X}_I}, d)$, where I corresponds to the reduct under consideration.

3.4 Aggregation operator methods

AGGREGATION OPERATOR.

Notions and conventions. An **aggregation operator** is a bounded monotone n-ary function $A : (X = \prod_{i=1}^n X_i, \leq_X) \rightarrow (Y, \leq_Y)$. For example the minimum and maximum operator, or the weighted sum with weights that sum up to one.
More flexible aggregation operators exist, like the Choquet integral [30], or its ordinal counterpart, the Sugeno integral [112]. It would lead us too far to explain these integrals in detail. It is enough to know that the Choquet integral can be characterised very nicely [75], its properties are well known, that it is capable of modelling several kinds of interaction like redundancy and synergy, and that it is an extension of the minimum and maximum operator, and of the weighted mean and the ordered weighted mean [125]. These integrals have $2^n - 2$ parameters, but by considering so-called k-additive Choquet/k-maxitive Sugeno integrals [26, 49], at most $\sum_{i=1}^k \binom{n}{i}$ parameters must be set. It should be noted that, as is the case for the weighted sum, these parameters can be interpreted in the context of game theory.
There exist plenty of other types of aggregation operators. Yet, it must be mentioned that in most cases, they demand the commensurability[17] of the axes to be aggregated, i.e. they are of the form $A : [0,1]^n \rightarrow [0,1]$, and mostly, they are idempotent, i.e. $A(x,\ldots,x) = x$. This makes that in general, aggregation operators can not be deployed immediately as discriminant functions in supervised learning. Some pre- and post-processing is usually needed, as described in Section 3.5.1.

3.4.1 The approach of Verkeyn et al.

In [116, 117], a specific data set from a large 3-fold survey on annoyance (noise, odour and light, with the number of criteria ranging from 12 to 23, and with a total of 3277 participants) is considered. The data is non-monotone, but in order to apply the proposed method, it must first be made monotone.
The considered data set has the characteristics that all criteria take values in the set $\{1,\ldots,5\}$, and also $\mathcal{L} = \{1,\ldots,5\}$. Of course, this does not mean that these

[17]It is possible to compare values from one axis to values from another axis.

values are commensurable, but in these papers, this assumption is implicitly made by mapping these values equidistantly onto $[0, 1] \subset \mathbb{R}$. The same for the labels. In fact, both pre- and post-processing steps as described in Section 3.5.1 are applied. They only consider 1 and 2-additive/maxitive integrals, the optimal parameter selection is done by a genetic algorithm. In [116], they try out Choquet integrals, in [117] Sugeno integrals are considered.
Only a comparison with the strongest component method (which is in fact just taking the maximum) is done, which is outperformed. Moreover, only the description of the data is aimed at.

3.4.2 The approach of Roubens et al., TOMASO

TOMASO, Tool for Ordinal Multi-Attribute Sorting and Ordering [76, 96] is also a very recent (2001-2002) method for learning a monotone ranking from a monotone learning sample based on the Choquet integral. There are however quite some differences with the previous approach. Instead of a direct aggregation, Roubens proposed in [96] to first transform the partial evaluations in a meaningful way into commensurable scales, in fact, he provides a technique to transform a ranking problem into a continuous aggregation problem. Then the parameters of a k-additive Choquet integral are assessed by translating the problem into a linear constraint satisfaction problem. First $k = 1$ is tried out, and k is increased until a satisfactory solution is found.

Commensurable scales. For each criterion c_i, define the *partial net score* as

$$S_i(a) := |(\mathbf{a}]_{\mathcal{X}_i}| - |[\mathbf{a})_{\mathcal{X}_i}| \ ,$$

for all $a \in \mathcal{S}$, i.e. the difference of the number of objects with a lower[18] partial score than a on c_i and the number of objects with a higher partial score on c_i. It holds that $c_i(a) \leq_{c_i} c_i(b) \iff S_i(a) \leq S_i(b)$. These scores are then normalised to

$$S_i^N(a) := \frac{S_i(a) + |\mathcal{S}| - 1}{2\,|\mathcal{S}| - 1} \in [0, 1] \,.$$

They can now be put together into a vector $S^N(a) = (S_1^N(a), \ldots, S_n^N(a))$.

Assignment of labels. Within the set of sample objects $\mathcal{S}$, denote the set of examples labelled i by $\mathcal{S}_i = \{a \in \mathcal{S} \mid d(a) = i\}$. In each of these sets, the subsets P_o and P_w of Pareto-optimal and Pareto-worst elements can be determined:

$$\begin{aligned} P_o(i) &= \{a \in \mathcal{S}_i \mid (\nexists b \in \mathcal{S}_i)(\mathbf{b} \geq_{\mathcal{X}} \mathbf{a})\} \,, \\ P_w(i) &= \{a \in \mathcal{S}_i \mid (\nexists b \in \mathcal{S}_i)(\mathbf{b} \leq_{\mathcal{X}} \mathbf{a})\} \,. \end{aligned}$$

[18]We can say “lower” instead of at most as high because of the subtraction.

These sets enable us to fix the boundaries in $[0,1]$ that enable the discriminant function to assign objects to classes. If A denotes the Choquet integral determined as explained above, then the lower and upper boundaries for class i are set as

$$\ell_i := \min_{b \in P_w(i)} A(S^N(b)) \quad \text{and} \quad r_i := \max_{b \in P_o(i)} A(S^N(b))\,.$$

Since the linear constraint satisfaction problem for determining the Choquet integral is devised in such a way that it always holds that $r_{i-1} < \ell_i$ (in fact, the distance between r_{i-1} and ℓ_i is maximised), the labelling of a new instance a is done as follows:

$$\lambda_{\text{TOMASO}}(a) = \begin{cases} i & \text{, if } A(S^N(a)) \in [\ell_i, r_i] \\ \{i-1, i\} & \text{, if } A(S^N(a)) \in \,]r_{i-1}, \ell_i[\,. \end{cases}$$

Remarks. The authors themselves remark that the method can not always guarantee a solution because the given problem may be incompatible with the assumption that the discriminant function is a Choquet integral. Unfortunately, they did not mention whether this situation is rare or not. If such a situation occurs, the user of the software is demanded to adapt the data set by revising the definition of the classes. It is however not mentioned how this can or should be done.
It should also be remarked that this approach bears some resemblances with support vector machines [25, 32]: both construct surfaces separating the classes via a linear constraint optimisation problem. Both only consider support vectors (in TOMASO, these are defined by the sets $P_o(i)$ and $P_w(i)$) to define this optimisation problem. This might be an indication of how the TOMASO approach could be extended to deal with non-separable (i.e. non-monotone) data by introducing positive slack variables in the constraints, turning the linear programming problem into a quadratic one.
Comparison test with other approaches have not yet been undertaken.

3.5 Related methods: ordinal classification and regression

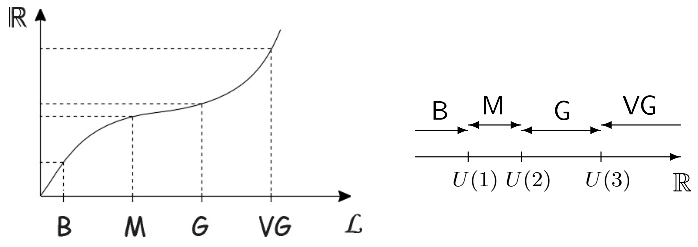
There have been quite some efforts concerning the non-monotone version of the supervised learning problem with ordinal class labels. Depending on the point of view, this problem can be denominated as ordinal classification, where an order is added to the class labels, or ordinal regression, where the numeric properties of the labels are eliminated. Another distinction between these two names is that in ordinal classification, the ordinal scale is handled as it is, while ordinal regression makes the assumption that there is some latent continuous variable underlying the ordinal variable.
It is also worth noticing that statistics is already active on this field for more than two decades, while the interest of the machine learning community only really emerged a couple of years ago.

3.5.1 Basic approach

In many situations, it is assumed that an ordinal variable is the result of a coarsely measured latent continuous variable. This is of course a quite rash assumption, which might be true in some cases, but certainly not always.
If this approach is chosen nonetheless, then two steps can be taken:

(i) pre-processing: transforming the ordinal variable into a continuous one, see Figure 3.8(a),

(ii) post-processing: transforming the final continuous result back into a value from the ordinal scale of class labels, see Figure 3.8(b).



(a) Pre-processing. (b) Post-processing

Figure 3.8: Examples of pre- and post-processing.

Pre-processing. In some old handbooks, you can still find prescriptions for transformations from ordinal to continuous variables. Of course, there does not exist such a holy recipe that merely has to be followed. Instead, these transformations should be dealt with case by case. One of the more advanced interactive methods available nowadays is probably a procedure used in the decision aid tool MacBeth, Measuring Attractiveness by a Categorical Based Evaluation Technique [5]. This procedure interactively builds a transformation of the ordinal scale into an interval or even a ratio scale.
It should be mentioned that, even if the ordinal scale is denoted by values such as $\{1, 2, \ldots, k\}$, using these values immediately in a continuous framework means that some transformation has been executed. So there is always a form of pre-processing (together with all its consequences), even if it is only implicitly.

Post-processing. Once the ordinal classification problem is altered to a *continuous* problem, i.e. a regression, any regression procedure can be applied. Since the outcome will be a real number, this number has to be transformed back into one of the values of the ordinal scale. Denoting this scale by $\{1, \ldots, k\}$, this means that a strictly increasing (extended) real function $U : \{0, 1, \ldots, k\} \to \overline{\mathbb{R}}$ with $U(0) = -\infty$ and $U(k) = +\infty$ needs to be defined. Any value in $]U(\ell - 1), U(\ell)]$ is then transformed into the label $\ell \in \{1, \ldots, k\}$.

Clearly, if the ordinal scale $\{1, \ldots, k\}$ would be used as such, then the most logical choice would be to round the result of the regression to the nearest value in $\{1, \ldots, k\}$, i.e. $U(\ell) = (2\ell + 1)/2$ for $\ell \in \{1, \ldots, k-1\}$.

Examples. An example of this approach from the recent literature (2001) is the application on the regression tree S-CART [66]. S-CART, Structural Classification and Regression Trees [67], is a fully relational version (based on propositional logic) of CART [23].
In [44], we can find this approach at work in an OSL (Ordinary Least Square) regression. This article demonstrates that this approach is common practice, even if, from a strict mathematical point of view, a transformation from a strictly ordinal scale into a richer numerical scale has no meaning.

3.5.2 Cumulative models

In statistics, the ordinal regression problem was tackled by the pioneering work of McCullagh in 1980 [77]. Bender and Benner [13] discuss how to use the most popular up-to-date ordinal regression techniques.
Most models are essentially based on the cumulative grouping of classes. If Y is the ordinal response variable taking values in $\{1, \ldots, k\}$, then the grouped classes $Y \geq i$ (or $Y \leq i$) are at the core of the ordinal regression models. Following [13], the two most popular approaches can be distinguished based on the kind of probabilities used in the generalised linear model [78]:

- *Grouped continuous models* based on probabilities $\mathcal{P}(Y \geq j \mid X = \mathbf{x})$ for $j = 2, \ldots, k$. (This group contains the proportional odds model.)

- *Continuation ratio models* based on probabilities[19] $\mathcal{P}(Y = j \mid Y \geq j, X = \mathbf{x})$ for $j = 2, \ldots, k$.

An alternative model that does not use a cumulative approach is the *adjacent categories model* based on probabilities $\mathcal{P}(Y = j+1 \mid Y = j, X = \mathbf{x})$ [2].

Remarks.
THE LATENT VARIABLE. These models all "*share the property that the categories can be thought of as contiguous intervals on some continuous scale. They differ in their assumptions concerning the distributions of the latent variable [...]. It may be objected, in a particular example, that there is no sensible latent variable and that these models are therefore irrelevant or unrealistic. However, the models* [referring to the proportional odds model and the proportional hazards model] *make no reference to the existence of such a latent variable and its existence is not required for model interpretation.* [77]"

[19]Or on probabilities $\mathcal{P}(Y = j \mid Y \leq j, X = \mathbf{x})$. It must be remarked that these two types of probabilities yield different results.

STOCHASTIC ORDERING. The *grouped continuous models* make the assumption that the data space $\mathcal{X}$ is stochastically ordered, i.e. for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$ it holds that

$$\mathcal{P}(Y \geq j \mid X = \mathbf{x}_1) \geq \mathcal{P}(Y \geq j \mid X = \mathbf{x}_2) \quad \text{for all } j \in \mathcal{L}\,, \quad \text{or}$$
$$\mathcal{P}(Y \geq j \mid X = \mathbf{x}_1) \leq \mathcal{P}(Y \geq j \mid X = \mathbf{x}_2) \quad \text{for all } j \in \mathcal{L}\,.$$

It is clear that this assumption is rather strong. Indeed the following configuration shown in Table 3.4 is excluded by this assumption:

	$\mathcal{P}(1 \mid \mathbf{x})$	$\mathcal{P}(2 \mid \mathbf{x})$	$\mathcal{P}(3 \mid \mathbf{x})$
$\mathbf{x}_1$	0.2	0.6	0.2
$\mathbf{x}_2$	0.1	0.9	0.1

Table 3.4: Impossible configuration in stochastically order space.

Naive translation to machine learning. The idea of using such probabilities $\mathcal{P}(Y \geq j \mid X = \mathbf{x})$ was tried out very recently (2001) on a meta-heuristic for supervised learners in [42]. However, they train *independently* $k-1$ learners to estimate the $k-1$ *dependent* cumulative probabilities. As a result, negative probabilities may arise, which are simply put to zero (see the WEKA [124] implementation `weka.classifiers.meta.OrdinalClassClassifier`).

3.5.3 The approach of Herbrich, a distribution independent model

In 1998-2000, Herbrich developed a distribution independent model for ordinal regression [56, 57, 58], as a response on the cumulative grouped models with their distributional assumptions on the latent variable, and their assumption of the stochastic ordering of $\mathcal{X}$. His approach is linked with preference learning (but, as we did in Section 3.3.1, we will suppress any references to it.)

The basic idea. In classification, the Bayes-optimal model identifies the optimal classifier $\lambda^* : \Omega \to \mathcal{L}$ as the one defined by $\lambda^*(a) = \arg\max_{i \in \mathcal{L}} \mathcal{P}(Y = i \mid X = \mathbf{a})$, leading to the least misclassifications. Herbrich argues that for ordinal regression, the classifier λ^*_{pref} (where the subscript refers to "preference") that induces the smallest number of rank reversals should be aimed at.

The risk functional. The optimal Bayesian solution λ^* minimises the risk functional $R(f) = E[\ell(f(a), \lambda(a)) \mid a \in \Omega]$, where λ is the unknown actual classifier we try to discover, and ℓ is the loss function $\ell(\hat{\imath}, i) = 1$ if $\hat{\imath} = i$, and 0 otherwise. Denoting $\mathcal{L} = \{1, \ldots, k\}$, the parameterised risk functional proposed for ordinal regression is

$$R^s_{\text{pref}}(f) = E[\ell^s_{\text{pref}}(f(a), f(b), \lambda(a), \lambda(b)) \mid (a, b) \in \Omega^2]\,,$$

with

$$\ell^s_{\text{pref}}(\hat{\imath},\hat{\jmath},i,j) = \begin{cases} 1 & \text{, if } 0 < i-j \leq s \text{ and } \hat{\imath}-\hat{\jmath} \geq 0\,, \\ 1 & \text{, if } 0 < j-i \leq s \text{ and } \hat{\jmath}-\hat{\imath} \geq 0\,, \\ 0 & \text{, otherwise.} \end{cases}$$

The parameter s "*controls the amount of assumed variation of* $\mathcal{P}(Y = i \mid X = \mathbf{x})$. *This can be treated as an assumption on the concentration of this probability around a "true" rank* [56]". The relation with Kendall's τ is also furnished, as well as bounds on the induced empirical risk (i.e. the risk on a learning sample). The classifier λ^*_{pref} minimising R^s_{pref} is then proven to be characterised by

$$\begin{aligned} \lambda^*_{\text{pref}}(a) >_{\mathcal{L}} \lambda^*_{\text{pref}}(b) &\iff \mathcal{P}(\lambda(a) >_{\mathcal{L}} \lambda(b) \mid \mathbf{a},\mathbf{b}) > \mathcal{P}(\lambda(b) >_{\mathcal{L}} \lambda(a) \mid \mathbf{a},\mathbf{b})\,, \\ \lambda^*_{\text{pref}}(a) = \lambda^*_{\text{pref}}(b) &\iff \mathcal{P}(\lambda(a) >_{\mathcal{L}} \lambda(b) \mid \mathbf{a},\mathbf{b}) = \mathcal{P}(\lambda(b) >_{\mathcal{L}} \lambda(a) \mid \mathbf{a},\mathbf{b})\,, \end{aligned}$$

where

$$\begin{aligned} \mathcal{P}(\lambda(a) >_{\mathcal{L}} \lambda(b) \mid \mathbf{a},\mathbf{b}) &= \sum_{i=1}^{k} \mathcal{P}(i \mid \mathbf{a}) \sum_{j=i+1}^{k} \mathcal{P}(j \mid \mathbf{b})\,, \\ \mathcal{P}(\lambda(b) >_{\mathcal{L}} \lambda(a) \mid \mathbf{a},\mathbf{b}) &= \sum_{i=1}^{k} \mathcal{P}(i \mid \mathbf{a}) \sum_{j=1}^{i-1} \mathcal{P}(j \mid \mathbf{b})\,. \end{aligned}$$

Application. Based on a linear utility model[20], a large margin principle algorithm (i.e. a support vector machine) was built which incorporates the proposed risk functional.

3.5.4 Support vector machines

Several adaptations of support vector machines were recently developed. We already mentioned Ralph Herbrich's efforts on this terrain. Another approach suggested in 2002 by Amnon Shashua and Anat Levin can be found in [104].

Finally, to be complete we mention the work of Thorsten Joachims [61]. He constructs a support vector machine for learning rankings. However, here the problem is how to learn a ranking from a given set of rankings, i.e. the examples are of the form $\langle \mathbf{x}, R\rangle$, where R is a weak order relation and the goal is to match a weak order to any object $a \in \Omega$. An example of such a problem is the ranking R of documents in order of relevance based on query $\mathbf{x}$. There can be found more literature on this subject, in different domains, but this is out of our present scope.

[20]See http://cepa.newschool.edu/het/essays/uncert/choiceref.htm, for a selection of references on utility theory.

3.5.5 Conclusion

The problem of supervised learning in a (monotone or not) ordinal context is still a freshly opened playground. Not that many methods have been developed yet, and there was made even less effort in investigating their theoretical frameworks, or in simply comparing the methods.
The distinct problem of monotone classification or monotone regression totally lacks any framework. All proposed solutions are very specific. Moreover, only four methods, namely OLM, MID, DBRS and OO are capable of dealing with the reality of non-monotone learning samples, but they are not all capable of rendering a monotone classifier.

APPENDIX

3.A Cornering (Makino/Potharst)

Note: This section of the appendix is more easily understood after reading Chapter 7.

During our experiments, it became clear that MDT, Monotone Trees [89] performs very good on classification accuracy, Kendall's tau, MAE and MSE, e.g. the experiments in Section 5.4.5 (see p. 135), and Section 7.5.3 (see p. 210). However, it also leads to rather large trees as is shown in the experiments of Appendix 3.B and in Table 7.8 (see p. 213).
In this section of the appendix, we have a closer look at the main engine of MDT, a process called "cornering". The idea is to add the corner elements (i.e. the extrema) of the intervals delineated by the leaves to the training data using some adequately labelling (see Section 3.2.3, p. 62). We show that this strategy is not always able to avoid the problem of so-called *blind splits* (see p. 200).
Consider the table shown in Figure 3.9(a), and suppose that, for some reason, the first split in the tree growing phase was determined as $c_1 \leq 1$. It is shown in Section 7.3.3, more specifically the paragraph entitled "Blind splits" (see p. 200), that a standard classification tree algorithm is unable to find the most appropriate split to obtain a monotone tree. However, if we apply the cornering method, then the training data is adapted before calculating the splitting measure: the corner elements to be added are shown in Figure 3.9(d). This means that the partitions pictured in Figure 3.9(c) are transformed into the ones shown in Figure 3.9(e). Obviously, this highlights the split based on c_3 as the preferred choice.
However, consider now the situation in Figure 3.10, Table (a), again with first split forced to be $c_1 \leq 1$. This time, cornering does no longer provide a full proof answer: it would lead to the splits $c_2 \leq 1$ on t_1, and $c_2 \leq 2$ on t_2. Necessitating at least one other split to obtain a monotone tree.

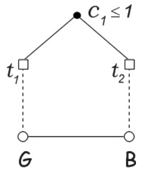
3.B Tree size of MDT.

This section of the appendix harbours some experimental results concerning the tree size (number of leaves) of MDT [89] compared to the tree size of C4.5 (both pruned and unpruned). Several designs (see Table 3.5) are considered, varying different parameters to gauge the effect they have on the tree size. The results are depicted in Table 3.6, they are the means of the number of runs as mentioned in the last column. In Appendix 3.A (see p. 80), we try to explain these results.

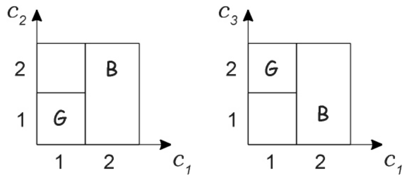
(a) Table.

	c_1	c_2	c_3	d
a_1	2	2	1	B
a_2	1	1	2	G

(b) Forced tree.



(c) Projection of the induced partitions for the two possible splits of t_1.



(d) Added corner elements.

	c_1	c_2	c_3	d
$a(t_1)$	1	1	1	B
$b(t_1)$	1	2	2	G
$a(t_2)$	2	1	1	B
$b(t_2)$	2	2	2	G

(e) Induced partitions after cornering.

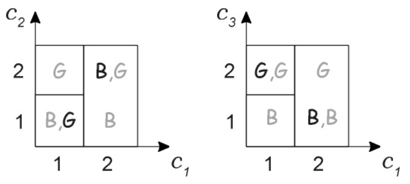


Figure 3.9: Cornering and blind splits (1).

(a) Table.

	c_1	c_2	c_3	d
a_1	2	2	1	B
a_2	1	3	2	G

(b) Induced partitions after cornering.

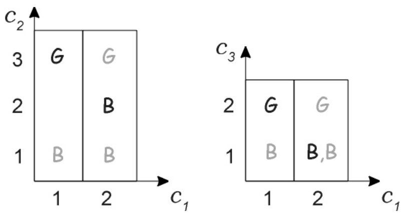


Figure 3.10: Cornering and blind splits (2).

	design 1	design 2	design 3	design 4
$\lvert C \rvert$ (# generated criteria)	$*$	8	*	7
$\lvert \mathcal{X}_c \rvert$ (# criterion values)	5	4	2	$*$
$\lvert \mathcal{L} \rvert$ (# labels)	4	$*$	2	4
$\lvert \mathcal{S} \rvert$ (# learning instances)	$*$	100	100	100

Table 3.5: Characteristics of the different designs.

(a) design 1

$\lvert C \rvert$	$\lvert \mathcal{S} \rvert$	C4.5 pruned	C4.5	MDT	runs
4	100	11±3.62	22.55±3.2	57.28±12.72	40
5	100	9.91±3.9	25.84±4.09	150.28±43.13	32
6	100	9.98±3.29	28.3±3.9	536.5±266.71	40
6	200	21.33	53.07	986.4	15

(b) design 2

labels	C4.5 pruned	C4.5	MDT	runs
2	8.9	16.4	1579	10
3	9.1	29.8	2994	11
4	6.9	30.3	1555	7

(c) design 3

$\lvert C \rvert$	C4.5 pruned	C4.5	MDT	runs
8	10	14.1	27.3	10
9	7.6	12.8	35.3	10
10	6.2	12	62	10
11	7.9	12.8	108	10
12	8.2	14.4	150	10
13	9.2	16.4	248	10
14	6.3	15.4	520	10
15	8.5	17.2	1025	10
16	7.4	15.9	1110	10
17	8.6	18.6	2377	10

(d) design 4

values	C4.5 pruned	C4.5	MDT	runs
2	15.1	24.4	42.1	10
3	12.5	29	225	10
4	11.5	32.6	1040	10
5	10.5	36.5	4715	10
6	9.3	35.2	10037	10

Table 3.6: Comparison of the tree size.

Interlude

GENESIS

The next chapter has probably the longest genesis of all, spread out over nearly three and a half years, from early autumn 2000 when I first started to work on the ranking problem, to the winter of 2003. Along the way, this manuscript was subject to some drastic changes, both in layout and contents. These changes were mostly inspired by the many referee reports I received on it, some of them full of praise, others cataloguing it as rubbish. I will let you decide for yourself.

Chapter 4

A framework for ranking: elementary granulation

In this chapter, we discuss how a proper definition of a ranking can be introduced in the framework of supervised learning. We elaborate on its practical representation, and show how we can deal in a sound way with reversed preferences by transforming them into uncertainties within the representation.

4.1 Introduction

4.1.1 Aims

We finally submerge in the profound and largely unexplored depths of ordinal ranking problems. Lighted only by the beacon of classification, and with the help of the torch of multi-criteria decision aid in our hands, we try to create some clarity in this vast jungle of classes, orders and preferences.

Supervised learning has been studied by many research groups, largely coming from statistics, machine learning and information (systems) science. In these studies, the problems of classification (discrete) and regression (continuous) have received a lot of attention. More recently the problem of ranking gained at interest because of the wide variety of applications it can be used for.

Ranking can be interpreted as monotone classification or monotone regression. The addition of the word "monotone" to the definition is, however, less trivial than it seems. And the problems this addition to the definition entails in the mathematical model used to deal with classification or regression are even more persistent. In this chapter (and in this thesis in general), we will mainly focus on discrete models, in other words, we will discuss how we can deal with monotone classification, which is equivalent to monotone ordinal regression.

Compared to classification, methods for solving ranking problems are only beginning to emerge. The reason for this slow progress lies partly in the fact that a solid framework for dealing with rankings in the supervised learning context has not yet been developed. For example, not all methods can guarantee that the generated classifier behaves monotonically, e.g. [10, 90]. Developing such a framework is the main goal set forth in this chapter. To be more precise, the aim is to develop a representation that can not be refuted based on semantical considerations. This is a very important aspect for any tool in decision aid: if the proposed solutions are not in line with the "common sense" (including semantical considerations such as monotonicity) of the user, the user will not accept the tool, even if it would deliver good performance on all kind of measures.

Another bare terrain in machine learning is that of working with ordinal data, e.g. [42, 57, 104], even though *"In measurement theory the raw data are typically postulated to be ordinal in nature* [72]". Mostly *"...the question addressed is the conditions under which the data structure exhibits numerical measures having certain properties"*. However, we prefer to work directly on the ordinal nature of the data itself, clearing ourselves of any conditions to be met.

4.1.2 Problems with earlier proposals

The aim of supervised learning is to discover a function $\lambda : \Omega \rightarrow \mathcal{L}$ based on a finite set of example pairs $(a, \lambda(a))$ with $a \in \Omega$. If $\mathcal{L}$ is finite, then λ is referred to as a classification. Generally, the objects $a \in \Omega$ are described by means of a finite set $Q = \{q_1, \ldots, q_n\}$ of attributes $q : \Omega \rightarrow \mathcal{X}_q$. Therefore, to each $a \in \Omega$ corresponds a vector $\mathbf{a} = (q_1(a), \ldots, q_n(a)) \in \mathcal{X} = \prod_{q \in Q} \mathcal{X}_q$ (called the **data space**, also known as measurement space), and the problem is then restated as learning the function λ based on examples $(\mathbf{a}, \lambda(a))$ with $a \in \Omega$. Although this new definition is not less restrictive if handled with care, it does tend to encourage a more narrow view, where $\lambda(a)$ is interpreted as $\lambda(\mathbf{a})$. This may lead to conflicting situations, since it is possible that $a, b \in \Omega$, $\mathbf{a} = \mathbf{b}$ but $\lambda(a) \neq \lambda(b)$. We will use the term *doubt* to refer to such a situation.

DATA SPACE.

The problem of ranking is generally formulated as a classification problem in the narrow view, with the additional restriction that it has to be monotone, i.e. for all vectors $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ we must have that $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ implies $\lambda(\mathbf{x}) \leq_{\mathcal{L}} \lambda(\mathbf{y})$, where $(x_1, \ldots, x_n) \leq_{\mathcal{X}} (y_1, \ldots, y_n)$ if and only if $x_i \leq_{\mathcal{X}_{q_i}} y_i$ for $i = 1, \ldots, n$, and the relations $\leq_{\mathcal{X}_q}$ on $\mathcal{X}_q$ and $\leq_{\mathcal{L}}$ on $\mathcal{L}$ are complete orders. Again, conflicting situations may arise, which we will refer to as *reversed preference* (see Section 4.5). Some authors [74, 89] impose some additional restrictions, such as demanding the training data to fulfill the monotonicity requirement, to ensure that these conflicts do not occur. Others [9] propose a form of naive conflict resolution.

However, a fundamental flaw in this definition is that it is formulated as a restriction not on the original definition $\lambda : \Omega \rightarrow \mathcal{L}$ of a classification, but on its operationalisation $\lambda : \mathcal{X} \rightarrow \mathcal{L}$, which was introduced in function of the description of the objects. Yet another problem is that ranking is not merely a restriction, but can also be seen as a generalisation of classification, in which the equality relation is replaced by an order relation. For example, in the formulation of a ranking given above, we see that "$\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ implies $\lambda(\mathbf{x}) \leq_{\mathcal{L}} \lambda(\mathbf{y})$" is an extension of "$\mathbf{x} = \mathbf{y}$ implies $\lambda(\mathbf{x}) = \lambda(\mathbf{y})$". It is well known that different points of view on a basic definition may lead to completely different extensions, so, if possible, the most intrinsic definition should be chosen. In our case, this means that $\lambda : \Omega \rightarrow \mathcal{L}$ is preferred.

It should be remarked that, while we focus in the first part of this chapter on a set-based representation of λ, there exists another strategy using cumulative models coming from statistics (e.g. cumulative logit model [2]) and later on applied in other learning schemes [42, 48, 53]. Instead of λ, it considers a family $(\lambda_i)_{i \in \mathcal{L}}$, where the $\lambda_i : \Omega \rightarrow \{0, 1\}$ are defined by $\lambda_i(a) = 0$ if $\lambda(a) \leq_{\mathcal{L}} i$, and 1 otherwise, in other words, the classes are grouped in sets of the form $\{\ell \leq_{\mathcal{L}} i \mid \ell \in \mathcal{L}\}$. In the last two sections of this chapter, we will incorporate this cumulative model (a probabilistic one) to construct a probabilistic representation of λ.

4.2 Classification and ranking

Notions and conventions. We will begin with some notions from lattice theory [19] and continue by giving a short introduction to preference modelling (see e.g. [41, 86, 118]).

As always in this thesis, we only consider binary relations. A relation R on X is said to be **complete** if for all $a, b \in X$ it holds that either aRb or bRa (or both). COMPLETE.

An **order (relation)** $\leq$ on a set X is a binary relation that is *reflexive* ($a \geq a$), *antisymmetric* (if $a \geq b$ and $b \geq a$, then $a = b$) and *transitive* (if $a \geq b$ and $b \geq c$, then $a \geq c$). As usual, the order $\leq$ decomposes into a *strict order* $<$ and an equality relation $=$. The couple $(X, \leq)$ is called a **poset** (partially ordered set). If neither $a \geq b$ nor $b \geq a$, we write $a \parallel b$ and call a and b **incomparable**. A **chain** is a poset without incomparable elements. Remark that in the latter case, the order $\geq$ is a complete order. ORDER (RELATION). POSET. INCOMPARABLE. CHAIN.

A **weak order** $\prec$ on X is a relation that is *asymmetrical* ($a \prec b$ implies not $b \prec a$) and *negatively transitive* (for all c, if $a \prec b$ then $a \prec c$ or $c \prec b$)[1]. A strict order is in particular a weak order. WEAK ORDER.

A **weak preference relation** (also called a **large preference relation** [118] S is a reflexive relation where the expression aSb stands for "a is at least as good as b" (it is also said that "a **outranks** b"). A weak preference relation can be decomposed into (and is totally defined by) three mutually exclusive relations: an asymmetric relation (**strict preference relation**) P with aPb if and only if aSb and not bSa, a reflexive and symmetric relation (**indifference relation**) I with aIb if and only if aSb and bSa, and an irreflexive and symmetric relation (**incomparability relation**) J with aJb if and only if not aSb and not bSa. Remark that P, I and J can be seen as generalisations of $<, =$ and $\parallel$. WEAK PREFERENCE RELATION. OUTRANKS. STRICT PREFERENCE RELATION. INDIFFERENCE RELATION. INCOMPARABILITY RELATION.

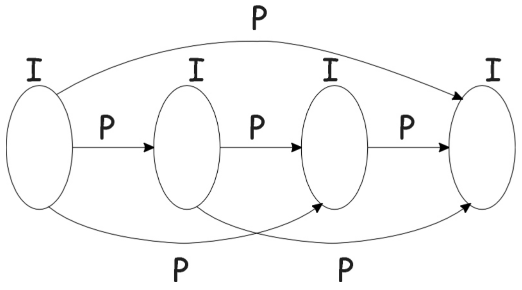
A **preorder** is a reflexive and transitive relation. In other words, it has the same properties as an order, except for antisymmetry, whence the name preorder. In this thesis, we will only consider complete preorders S. Given such an S, we can interpret it as a weak preference relation (see Figure 4.1). We find $S = P \cup I$, with both P and I transitive. Remark that in [86] it is shown that S is a complete preorder if and only if P is a weak order. PREORDER.

Recapitulation Section 2.2

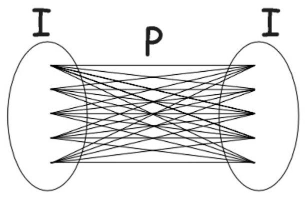
4.2.1 Classification

A **classification** λ in Ω is defined as the assignment of the objects belonging to Ω, to some element, called a **class label**, in a universe $\mathcal{L}$ of labels. If $\mathcal{L}$ is a continuum, λ is usually referred to as a **regression**. The class labels can be identified with their inverse image in the object space Ω, where they constitute a partition. We will call these inverse images **(object) classes**. For any class label $i \in \mathcal{L}$, we denote the corresponding class by $C_i := \lambda^{-1}(i)$, and $\mathrm{Cl} := \{C_i \mid i \in \mathcal{L}\}$. So, the set of all classifications in Ω stands in one-to-one correspondence to the set of all partitions of Ω (which is equivalent to the set of all equivalence relations on Ω). Remark that

[1]Remark that negative transitivity and asymmetry imply transitivity.



(a) A complete preorder S consist of an equivalence relation I and a weak order P that induces a strict complete order on the equivalence classes.



(b) Detail of the corresponding weak order P.

Figure 4.1: Visual representations

we may assume that λ is surjective by constraining $\mathcal{L}$ to the image of λ (this can be done without loss of generality since by definition, we are only interested in objects from Ω, which may well be infinite).

4.2.2 Ranking

Preliminaries. If we want to define a ranking based on the above definition of a classification, it becomes clear that there is no room for a concept such as monotonicity since Ω has no inherent structure such as $\mathcal{X}$. Still, we have to plant the seeds for it, such that monotonicity will appear naturally when the data space $\mathcal{X}$ is introduced as a representation of Ω. This can be done, following the ideas from [53, 97], by returning to the semantics behind ranking, which declares that the higher an object's rank, the more it is preferred. We can model this preferential information by a complete preorder.

Definition. In the MCDA (Multi-Criteria Decision Analysis) literature [97], the term "sorting" is used to refer to a classification into a pre-defined finite set of ordered classes. The best way to enlighten the meaning of words is to turn to a dictionary. According to Webster's Encyclopedia Unabridged Dictionary [122], the meaning of "sort" is: "n. a particular kind, species, variety, class, or group, distinguished by a common character or nature; v.t. to arrange according to sort, kind or class". On the other hand, looking up the word "rank" results in: "n. a number of persons forming a separate class in a social hierarchy or in any graded body; relative position or standing; v.t. to arrange in ranks or in regular formation". Therefore, we feel the term "ranking" would be better suited. More general, we define a ranking as:

Definition 4.2.1

RANKING.

A **ranking** in Ω is a classification/regression $\lambda : \Omega \to \mathcal{L}$, together with an order $\geq_{\mathcal{L}}$ on $\mathcal{L}$. We denote this ranking by $(\lambda, \geq_{\mathcal{L}})$. Moreover, the order $\geq_{\mathcal{L}}$ defines a weak preference relation S on Ω as follows:

$$aSb \iff \lambda(a) \geq_{\mathcal{L}} \lambda(b)\,,$$

or stated differently, aSb if and only if $s \geq_{\mathcal{L}} r$, for any $a \in C_s$ and $b \in C_r$, with $C_i = \lambda^{-1}(i)$ denoting the class associated with the label $i \in \mathcal{L}$.

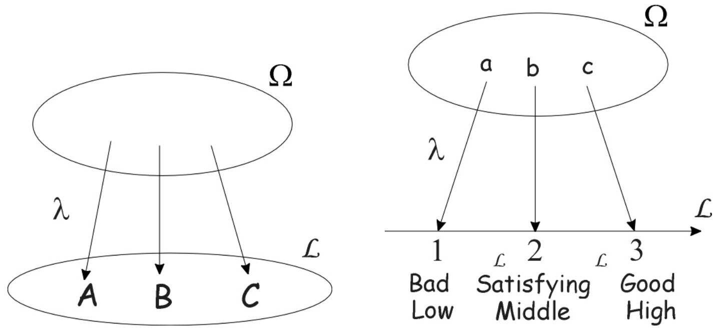
COMPLETE RANKING.

In this thesis, we will only consider **complete ranking**s, where $\geq_{\mathcal{L}}$ is a complete order on $\mathcal{L}$, i.e. $(\mathcal{L}, \geq_{\mathcal{L}})$ is a chain. This is in line with most of the current problems considered in supervised learning. In this case, we have a specific preference structure on Ω linked with the classes. For $a \in C_s$ and $b \in C_r$ it holds that

$$aPb \iff s >_{\mathcal{L}} r \qquad \text{and} \qquad aIb \iff s = r\,.$$

So, the set of all rankings in Ω stands in one-to-one correspondence to the set of all complete preorders S on Ω. The classes are formed by the indifference relation I which is an equivalence relation (transitivity holds for I since S is a complete preorder). Hence, the indifference relation I determines the classification.

Remarks. The foregoing definition of a ranking consists of two parts: first, a classification/regression with an ordered image; second, the associated semantics expressed by a weak preference relation. It is the second condition that ensures that a (finite) ranking is not simply an ordinal classification. The difference between a classification and a ranking is shown in Figure 4.2. A second point worth noting in



(a) Classification, possibly with $A \leq_{\mathcal{L}} B \leq_{\mathcal{L}} C$.

(b) Ranking, c preferred to b preferred to a

Figure 4.2: Classification and ranking

the definition is that the set $\mathcal{L}$ is not necessarily finite (the same is true in the definition of a classification). Still, in the remainder of this chapter, we will assume $\mathcal{L}$ to be finite. We will also assume that $\mathcal{X}$ is finite to obtain a pure ordinal setting.

Assumptions: $\mathcal{X}$ and $\mathcal{L}$ are finite, $(\mathcal{L}, \leq_{\mathcal{L}})$ is a chain.

People familiar with the internet usually link the term “ranking” with the results of a search query (e.g. using *Google*). In that case, a ranking is defined as a weak order on Ω (see e.g. [61]). This corresponds to the special case where $S = P$ (i.e. there are no indifferent objects because the relation is asymmetrical), and hence $|\mathcal{L}| = |\Omega|$.

4.3 Representing a classification: sets

Recapitulation Section 2.2

The above definitions are not really useful in practice since they relate to a universe Ω that is in essence just an enumeration of all the objects. To access some of the interesting properties of the objects, we fall back on a set of attributes Q. In this way, we can represent each object $a \in \Omega$ by a vector $\mathbf{a} = (q_1(a), \ldots, q_n(a)) \in \Omega_{\mathcal{X}} \subseteq \mathcal{X}$, where $\Omega_{\mathcal{X}}$ is the set of all measurement vectors corresponding to objects in Ω. This also leads to a representation $\hat{\lambda}$ of the classification $\lambda : \Omega \to \mathcal{L}$ in the following way:

$$\begin{array}{rcrcl} \hat{\lambda} & : & \Omega_{\mathcal{X}} & \to & 2^{\mathcal{L}}\,, \\ & & \mathbf{x} & \mapsto & \hat{\lambda}(\mathbf{x}) = \{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}\,, \end{array}$$

where $2^{\mathcal{L}}$ is the power set of $\mathcal{L}$, i.e. the set of all subsets of $\mathcal{L}$. Thus, the representation of a classification is again a classification, but now in the space $\Omega_{\mathcal{X}} \subseteq \mathcal{X}$, with classes $\hat{\lambda}^{-1}(I)$, where $I \subseteq \mathcal{L}$. We may also define a classification $\hat{\lambda}^* : \Omega \to 2^{\mathcal{L}}$ by setting[2] $\hat{\lambda}^*(a) = \hat{\lambda}(\mathbf{a})$. The classes of $\hat{\lambda}^*$ are denoted by $\mathbf{C}_I = (\hat{\lambda}^*)^{-1}(I)$, these are the so-called **decision regions** of λ.
Remark that $\Omega \cong \mathcal{X}$ (Ω is *isomorphic* to $\mathcal{X}$) implies that $\hat{\lambda} \cong \lambda$. This property of isomorphism states that the representation $\hat{\lambda}$ is a very natural one. We certainly want to keep this property in the case of rankings. Moreover, the more general observation that representing a classification results again into a classification is also a very desirable property. We know that the real problem is one of classification, but we will work with a representation, so it would be against our intuition that this representation would become something different from a classification. In the same line of thinking, we would like this property, if possible, to hold for rankings as well.

4.4 Representing a ranking: intervals

Notions and conventions. Let $(X, \leq)$ be a poset. The subset $[a, b] = \{x \in X \mid a \leq x \leq b\}$ is called a **(poset) interval** in $(X, \leq)$. We will also consider the **half open intervals** $(a] = \{x \in X \mid x \leq a\}$ and $[a) = \{x \in X \mid a \leq x\}$.

$[a, b]$
(POSET) INTERVAL.
HALF OPEN INTERVALS.
$(a], [a)$

[2]Since $\hat{\lambda}$ and $\hat{\lambda}^*$ both express the same idea, we will not restrain ourselves from mixing their usage.

In this paper, the term "order" is used in its strict mathematical sense (i.e. a reflexive, antisymmetric, transitive binary relation). The term "ordering", however, is used more freely, conveying a semantical idea rather than a mathematical one.

General notation. As mentioned above, we would like to have that if $\Omega \cong \mathcal{X}$, then a representation of a ranking $(\lambda, \geq_{\mathcal{L}})$ should be a ranking once again. therefore, we will denote a **representation of a ranking** by

REPRESENTATION OF A RANKING.

$$\boxed{(\lambda_{\text{repr}}, \trianglerighteq_{\text{Im}})}$$

where $\trianglerighteq_{\text{Im}}$ is some relation on the image of λ_{repr} that should be isomorphic to $\geq_{\mathcal{L}}$ if $\Omega \cong \mathcal{X}$.

4.4.1 The image of a ranking

Let $(\lambda, \geq_{\mathcal{L}})$ be a (complete) ranking. A first remark concerns the range of $\hat{\lambda}$ when we are dealing with rankings. Since λ stands for a classification, we could try to define $\hat{\lambda}$ as the representation of this classification as in Section 4.3. However, imagine we have three classes, labelled Bad, Moderate or Good. For $\mathbf{x} = (x_1, \ldots, x_n) \in \Omega_{\mathcal{X}}$, the assignment $\hat{\lambda}(\mathbf{x}) = \{\text{Bad, Good}\}$ means that taking into account the partial evaluations $(x_1, \ldots, x_n)$, $\mathbf{x}$ is globally evaluated as either Bad or Good, but never as Moderate, although in going from Bad to Good, one must pass through Moderate. So, the definition of the representation of λ does not make sense in the context of ranking. The solution for this example is obvious: redefine $\hat{\lambda}(\mathbf{x}) = \{\text{Bad, Moderate, Good}\}$.

Example 4.4.1. *Just think about an evaluation process: it would not make sense to tell somebody* "Based on the tests you passed so far, we must conclude you are either a Good or a Bad candidate, but obviously, you are not a Moderate one. We need you to undergo another test to get a definitive answer." *Do not confuse the previous with a situation like* "You are clearly a Good candidate. However, the final assessment showed that your manner of handling things does not stroke with our company's culture, so we are sorry to inform you that..." *Here the candidate is evaluated as having a tendency towards Good until the last test, where it is decided the candidate does not belong to the class of Good candidates.*

In general, it is no longer useful to consider the entire power set $2^{\mathcal{L}}$ as the range of $\hat{\lambda}$, because some assignments become meaningless as observed above. To get a better understanding, it is clarifying to think of the notions of half open intervals[3] in the chain $(\mathcal{L}, \geq_{\mathcal{L}})$. Resuming our example, we have $\mathcal{L} = \{\text{B(ad), M(oderate), G(ood)}\}$, and B $<_{\mathcal{L}}$ M $<_{\mathcal{L}}$ G. An assignment to one of these classes makes sense, for example $\hat{\lambda}(\mathbf{x}) = \{\text{B}\}$. It is possible to express $\{\text{B}\}$ in terms of intervals: $\{\text{B}\} = [\text{B}) \cap (\text{B}] = [\text{B}, \text{B}]$. An assignment to $\{\text{M,G}\}$ is also meaningful, and we have

[3]In the dominance-based rough sets approach (see [53]) these notions are linked with the so-called *upward* and *downward union* of classes.

{M,G} $=$ [M) $\cap$ (G] $=$ $[\mathrm{M},\mathrm{G}]$. However, an assignment such as {B,G} is not meaningful, and we have {B,G} $\neq [B,G] =$ {B,M,G}.
This leads to the conclusion that only intervals in $(\mathcal{L}, \leq_{\mathcal{L}})$ make sense as possible values for $\hat{\lambda}$. Indeed, if there is doubt in ranking $\mathbf{x} \in \Omega_{\mathcal{X}}$, in other words, if there are two different labels $r, s \in \mathcal{L}$ such that $\{r, s\} \subseteq \hat{\lambda}(\mathbf{x})$, then also the intermediate labels should belong to the assignment of $\mathbf{x}$, i.e. if $r \leq_{\mathcal{L}} s$ then $[r, s] \subseteq \hat{\lambda}(\mathbf{x})$. Remark that if $r >_{\mathcal{L}} s$, then always $[r, s] = \emptyset \subseteq \hat{\lambda}(\mathbf{x})$. Formally,

$$(\forall \mathbf{x} \in \Omega_{\mathcal{X}})(\forall (r, s) \in \mathcal{L}^2)(\{r, s\} \subseteq \hat{\lambda}(\mathbf{x}) \Rightarrow [r, s] \subseteq \hat{\lambda}(\mathbf{x}))\,.$$

This property characterises $\hat{\lambda}(\mathbf{x})$ as a(n) (order) convex subset of the lattice $(\mathcal{L}, \geq_{\mathcal{L}}$). Moreover, since $(\mathcal{L}, \geq_{\mathcal{L}})$ is a finite chain, this means that $\hat{\lambda}(\mathbf{x})$ is an interval in $(\mathcal{L}, \geq_{\mathcal{L}})$. Thus the range of $\hat{\lambda}$, i.e. the actual decision space, is no longer $2^{\mathcal{L}}$, but rather

$$\mathcal{L}^{[2]} = \{[r, s] \mid (r, s) \in \mathcal{L}^2 \wedge r \leq_{\mathcal{L}} s\}.$$

We may now define $\hat{\lambda}$ in a meaningful way as follows:

$$\boxed{\begin{array}{cccl} \hat{\lambda} & : & \Omega_{\mathcal{X}} & \rightarrow & \mathcal{L}^{[2]} = \{[r, r'] \mid (r, r') \in \mathcal{L}^2 \wedge r \leq_{\mathcal{L}} r'\}\,, \\ & & \mathbf{x} & \mapsto & [\hat{\lambda}_\ell(\mathbf{x}), \hat{\lambda}_r(\mathbf{x})]\,, \end{array}} \tag{4.4.1}$$

where

$$\begin{aligned} \hat{\lambda}_\ell(\mathbf{x}) &= \min\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}\,, \\ \hat{\lambda}_r(\mathbf{x}) &= \max\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}\,. \end{aligned}$$

Remark that if we write $I \in \mathcal{L}^{[2]}$ as an interval $[r, s]$, it holds by definition that

$$\mathbf{C}_{[r,s]} \neq \emptyset \implies (\exists a, b \in \mathbf{C}_{[r,s]})(a \in C_r \wedge b \in C_s)\,. \tag{4.4.2}$$

4.4.2 Ordering the image

Notions and conventions. An **interval order**[4] S on a finite set Ω is a reflexive and **Ferrers**[5], whence complete, relation. An equivalent formulation expresses an interval order S as a reflexive relation together with two functions l and r from Ω to some chain $(L, \geq)$ associating to each $a \in \Omega$ an interval $[l(a), r(a)]$ such that $aSb \Leftrightarrow r(a) \geq l(b)$.

INTERVAL ORDER.
FERRERS.

Conditions on the ordering. There are some intuitive conditions we want to impose on the relation $\trianglerighteq_{\text{Im}}$. It should be (i) reflexive, to ensure that equal elements of the image of λ_{repr} are also treated equally, (ii) an extension of $\geq_{\mathcal{L}}$, as discussed

[4]In the sense of [86].
[5]A relation R on Ω is called **Ferrers** if $(aRb \wedge cRd) \Rightarrow (aRd \vee cRb)$, for any $a, b, c, d \in \Omega$. .

previously, and (iii) meaningful, just as $\geq_{\mathcal{L}}$ has a meaning in terms of a preference relation. So, we would like to have an interpretation such as

$$\boxed{\lambda_{\text{repr}}(\mathbf{a}) \trianglerighteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{b}) \iff a\widehat{S}b} \tag{4.4.3}$$

where $a\widehat{S}b$ means: *based on the information derived from Q and λ_{repr}, we conclude that a is at least as good as b.* Remark that condition (i) has an additional advantage in this context, since it enables us to interpret $\trianglerighteq_{\text{Im}}$ as a weak preference relation.
Lastly, closely related to the third condition, we would like that (iv) $\trianglerighteq_{\text{Im}}$ does not depend on λ. In other words, the relation $\trianglerighteq_{\text{Im}}$ must be derived from $(\mathcal{L}, \geq_{\mathcal{L}})$. So, if we have two rankings $(\lambda_1, \geq_{\mathcal{L}})$ and $(\lambda_2, \geq_{\mathcal{L}})$, then their representations should be $(\hat{\lambda}_1, \trianglerighteq_{\text{Im}})$ and $(\hat{\lambda}_2, \trianglerighteq_{\text{Im}})$. This is a consequence of the following observation: the order on the label set is the most intricate part of a ranking, first we have an ordered label set, and afterwards we assign objects to these labels in accordance with the semantics behind this order.
Next, we study a representation continuing with $\lambda_{\text{repr}} = \hat{\lambda}$ in the spirit of the previous section, i.e. using the set interpretation (2.2.1). Later on, in Section 4.7, we will consider the distributional interpretation (2.2.3) for λ_{repr}.

First attempts. Starting from (ii) and (iii), we will try to extend the semantics behind $\geq_{\mathcal{L}}$. Recall that we have by definition of S derived from a ranking that

$$\begin{aligned} s \geq_{\mathcal{L}} r \quad &\iff \quad (\forall a \in C_s)(\forall b \in C_r)(aSb) \\ &\iff \quad (\exists a \in C_s)(\exists b \in C_r)(aSb)\,, \end{aligned}$$

where the second equivalence is due to the fact that the objects inside one class are all considered to be indifferent to each other.
These expressions can easily be generalised by replacing $s \geq_{\mathcal{L}} r$ by $I \trianglerighteq_{\text{Im}} I'$, where $I, I' \in \mathcal{L}^{[2]}$, and C_s (resp. C_r) by $\mathbf{C}_I \neq \emptyset$ (resp. by $\mathbf{C}_{I'} \neq \emptyset$):

$$I \trianglerighteq^1_{\text{Im}} I' \iff (\forall a \in \mathbf{C}_I)(\forall b \in \mathbf{C}_{I'})(aSb)\,, \tag{4.4.4}$$

and

$$I \trianglerighteq^2_{\text{Im}} I' \iff (\exists a \in \mathbf{C}_I)(\exists b \in \mathbf{C}_{I'})(aSb)\,. \tag{4.4.5}$$

If we impose reflexivity, and write $I = [s_1, s_2]$, $I' = [r_1, r_2]$, this finally results in (using Expression (4.4.2))

$$[s_1, s_2] \trianglerighteq^1_{\text{Im}} [r_1, r_2] \iff s_1 \geq_{\mathcal{L}} r_2\,,$$

for Expression (4.4.4), and

$$[s_1, s_2] \trianglerighteq^2_{\text{Im}} [r_1, r_2] \iff s_2 \geq_{\mathcal{L}} r_1\,,$$

for Expression (4.4.5), see also Figure 4.3. Relation $\trianglerighteq^1_{\text{Im}}$ is an order, and relation $\trianglerighteq^2_{\text{Im}}$ is an interval order.

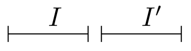
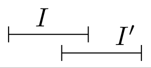
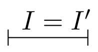
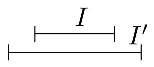
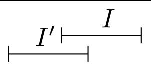
	$I \lhd^1_{\text{Im}} I'$	$I \lhd^2_{\text{Im}} I'$
	$I \parallel^1_{\text{Im}} I'$	$I \sim^2_{\text{Im}} I'$
	$I \sim^1_{\text{Im}} I'$	$I \sim^2_{\text{Im}} I'$
	$I \parallel^1_{\text{Im}} I'$	$I \sim^2_{\text{Im}} I'$
	$I \parallel^1_{\text{Im}} I'$	$I \sim^2_{\text{Im}} I'$

Figure 4.3: Visualisation of $\unlhd^1_{\text{Im}}$ and $\unlhd^2_{\text{Im}}$

It is clear that these two relations fulfill conditions (i), (ii) and (iv). However, in both cases there are some problems with condition (iii). We have for instance that $[1,3]$ and $[1,2]$ are incomparable w.r.t. $\unrhd^1_{\text{Im}}$. However, we would prefer an object with label $[1,3]$ over another with label $[1,2]$ if that is all we know about these objects. Indeed, if an object is assigned to $[1,2]$, it may belong to either of the classes C_1 and C_2, whereas an object assigned to $[1,3]$ may also belong to the better class C_3. So, for two objects $a, b \in \Omega$, knowing only $\hat{\lambda}(\mathbf{a}) = [1,3]$ and $\hat{\lambda}(\mathbf{b}) = [1,2]$, we would prefer a over b, implying that we would like to have $[1,3] \unrhd_{\text{Im}} [1,2]$. This reasoning is in line with the semantics (4.4.3) we are pursuing. Whereas $\unrhd^1_{\text{Im}}$ proves to be too restrictive, the relation $\unrhd^2_{\text{Im}}$ seems to be too permissive, being indifferent between $[1,2]$ and $[1,3]$. This means that we need to find something in between the generalisations of (4.4.4) and (4.4.5).

Second attempt. There exist quite some different kinds of orders that could be possible candidates for $\unrhd_{\text{Im}}$, see e.g. [40]. Unfortunately, their known characterisations are not particularly helpful in pinpointing one or more of them in the present context. We could simply check them all out and see whether they suit our purpose. But at this point, it is more interesting to let us guide by the reasoning demonstrated in the previous paragraph, and to first state the desired semantics and translate it afterwards into a suitable expression.
In that way we define $I \unrhd_{\text{Im}} I'$ if and only if I is an improvement over I' or I' is a deterioration compared to I. We will now translate this into mathematical expressions. Assuming that $\mathbf{C}_I$ and $\mathbf{C}_{I'}$ are non-empty, we say that I is an **improvement** of I' if and only if

IMPROVEMENT.

$$\begin{cases} (\exists a \in \mathbf{C}_I)(\forall b \in \mathbf{C}_{I'})(aSb) \\ (\forall a \in \mathbf{C}_I)(\exists b \in \mathbf{C}_{I'})(aSb)\,. \end{cases} \tag{4.4.6}$$

Likewise, I' is a **deterioration** of I if and only if

DETERIORATION.

$$\begin{cases} (\exists b \in \mathbf{C}_{I'})(\forall a \in \mathbf{C}_I)(aSb) \\ (\forall b \in \mathbf{C}_{I'})(\exists a \in \mathbf{C}_I)(aSb)\,. \end{cases} \tag{4.4.7}$$

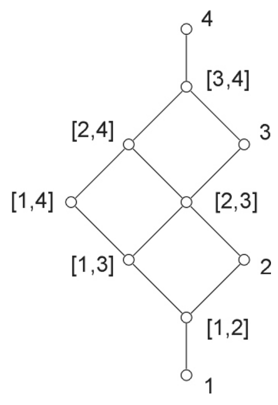


Figure 4.4: The order $\leq^{[2]}$ on $2^{\mathcal{L}}$ with $\mathcal{L} = \{1, 2, 3, 4\}$ (Hasse diagram).

It immediately strikes that all of these expressions can be seen as intermediate to the generalisations of (4.4.4) and (4.4.5). If we write $I = [s_1, s_2]$ and $I' = [r_1, r_2]$, it can be shown quite easily that (remember that $\mathbf{C}_I$ and $\mathbf{C}_{I'}$ are assumed to be non-empty)

$$(4.4.6) \iff (4.4.7) \iff ((r_1 \leq_{\mathcal{L}} s_1) \wedge (r_2 \leq_{\mathcal{L}} s_2)) \ .$$

Hence, we find that $\trianglerighteq_{\text{Im}}$ is an order, which we will denote by $\geq^{[2]}$, defined as follows:

Definition 4.4.1

Let $I, I' \in \mathcal{L}^{[2]}$. If we write $I = [r_1, r_2]$ and $I' = [s_1, s_2]$, we put

$$[r_1, r_2] \leq^{[2]} [s_1, s_2] \iff ((r_1 \leq_{\mathcal{L}} s_1) \wedge (r_2 \leq_{\mathcal{L}} s_2)) \ .$$

So, based on a semantical discourse, we arrived at the order $\leq^{[2]}$, which is already extensively studied and whose properties are well known [39], for example, it turns $(\mathcal{L}^{[2]}, \leq^{[2]})$ into a complete lattice[6]. An example on $\mathcal{L} = \{1, 2, 3, 4\}$ can be found in Figure 4.4.

Some afterthoughts. It is clear that now all four conditions are met. It should be noted that the order $\leq^{[2]}$ was derived from the premise that we only have access to intervals of values to reach a decision. If other information would be available, other orderings might prevail. For example, distributional information might lead to a stochastic ordering (see Section 4.7), or if risk aversion underlies the

[6]Potharst [89] also introduced this order in the setting of rankings, however, without being aware of its semantics in the context of ranking.

decision, then we could consider the leximin[7] order $\leq_1$. Consider for example $\mathcal{L} = \{1, 2, 3, 4\}$, then we have

$$1 \leq_1 [1,2] \leq_1 [1,3] \leq_1 [1,4] \leq_1 2 \leq_1 [2,3] \leq_1 [2,4] \leq_1 3 \leq_1 [3,4] \leq_1 4\,.$$

If risk would be favoured, the following order $\leq_2$ might be suitable:

$$1 \leq_2 [1,2] \leq_2 2 \leq_2 [1,3] \leq_2 [2,3] \leq_2 3 \leq_2 [1,4] \leq_2 [2,4] \leq_2 [3,4] \leq_2 4\,.$$

Note that both situations are in line with the order $\leq^{[2]}$ just defined ($\leq^{[2]} \subseteq \leq_i$). Even more, the order $\leq^{[2]}$ is nothing else but the intersection of $\leq_1$ and $\leq_2$. This also pleads for the non-invasive character of the order $\leq^{[2]}$ (no presuppositions about the preferences are imposed).

4.5 The monotonicity constraint

Up to now, we have given a definition of a (complete) ranking $(\lambda, \geq_{\mathcal{L}})$ and have shown a possible representation by the (not necessarily complete) ranking $(\hat{\lambda}, \geq^{[2]})$. Note that we did not need any form of monotonicity for the definition or this representation. Monotonicity will arise in a natural way when taking into account the attributes, or rather, the criteria (see below) used to describe the properties of objects.

4.5.1 Preliminaries

Let us first turn back to classifications. Even if we assume a classification λ to be deterministic in the sense that any object $a \in \Omega$ is assigned to exactly one class with label in $\mathcal{L}$, we still cannot guarantee that we have $|\hat{\lambda}^*(a)| = 1$ for all $a \in \Omega$. This is a consequence of the possible occurrence of *doubt* (called "inconsistency" in rough set theory [65, 109])

Definition 4.5.1 (see p. 32)

(i) There is **doubt** between the classification λ and the set of attributes Q if

$$(\exists (a,b) \in \Omega^2)(\mathbf{a} = \mathbf{b} \wedge \lambda(a) \neq \lambda(b))\,.$$

(ii) There is **doubt** inside the representation $\hat{\lambda}$ if

$$(\exists \mathbf{x} \in \Omega_{\mathcal{X}})(|\hat{\lambda}(\mathbf{x})| > 1)\,.$$

[7]The ordering as used in a dictionary.

It is clear that these two notions of doubt coincide. In the case of doubt, the function $\hat{\lambda}^*$ can assign a set of labels to an object, indicating that it is not possible to label the object with one specific class label based on the associated vector. Remark that the first definition emphasises the conflict between the vector representations of the objects and the classification λ (as discussed in Section 4.1.2), the second definition stresses the ensuing idea of uncertainty. Also note that if $\hat{\lambda}$ is interpreted as a classification, we have that there is never doubt between the classification $\hat{\lambda}$ and the (corresponding) set of attributes Q.

4.5.2 Multi-criteria decision aid (MCDA)

CRITERION.

TRUE CRITERIA.

In the context of ranking, the attributes have a specific interpretation, and are usually referred to as criteria. A **criterion** [97] is defined as a mapping $c : \Omega \to (\mathcal{X}_c, \geq_c)$, where $(\mathcal{X}_c, \geq_c)$ is a chain, such that it appears meaningful to compare two objects a and b, according to a particular point of view, on the sole basis of their evaluations $c(a)$ and $c(b)$. This means that a criterion induces a weak preference relation S_c on Ω. In this paper, we will only consider **true criteria** [20], where the induced weak preference relation is a complete preorder defined by $aS_cb \Leftrightarrow c(a) \geq_c c(b)$. We assume to have a finite set of criteria $C = \{c_1, \ldots, c_n\}$ at our disposal.

Example 4.5.1. *In Chapter 2, you did some classification of small animals like dogs, cats and rabbits. After being around all those wonderful furry creatures, you feel like taking in one yourself. So you do some very convincing pleading to your (girl/boy)friend/husband/wife, and soon it is agreed a pet is more than welcome, but: What kind of animal? It should certainly be not extremely big, not a horse or a cow or anything alike, it should be an animal you can take inside, any animal larger than 1,5 meter is not even an option (Bad). You prefer that it is not too small either, you don't want to come home a bit tired and accidentally step on your hamster. And your companion would rather prefer that it is not of such a size that it will trash your furniture whenever it is left alone for 5 minutes in your living room (Satisfying). The not too small animals and the not too big animals are welcome (Good), but you agree that ideally, the new house mate should be between 40 and 80 cm (Very Good).*
A second criterion you agreed upon was that the breed should not be known to bear dogs of a nervous nature, so no yappers or anything alike.

DOMINANCE RELATION.

If we assume that all criteria are true criteria (see Section 6.2.3 for a more complete discussion), then the **dominance relation**[9] $\triangleright_C$ on Ω w.r.t. C is defined by

$$a \triangleright_C b \iff \begin{cases} (\forall c \in C)(aS_cb) \\ (\exists c \in C)(aP_cb) \end{cases}$$

[8]For demonstration purposes, we have chosen an ordinal scale that corresponds to a coarsely measured numerical scale.

[9]In the literature, the dominance relation is usually denoted by Δ_C. Because of the symmetrical nature of the symbol Δ_C, we feel it does not clearly denote its meaning and prefer to use the notation $\triangleright_C$.

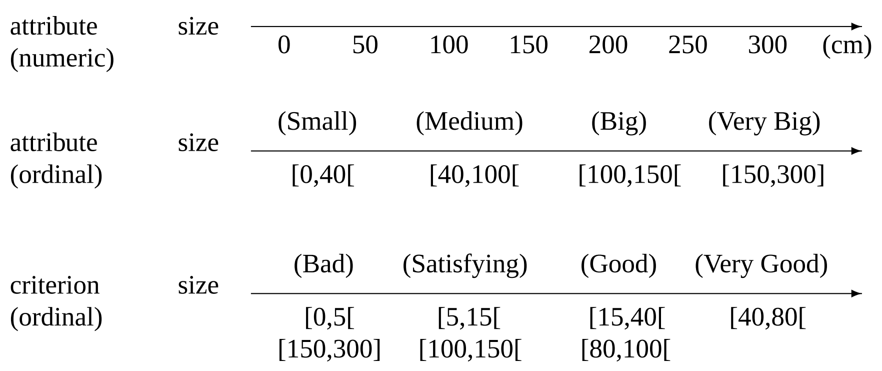


Figure 4.5: Difference between attribute and criterion[8].

for any $a, b \in \Omega$. It is said that a **dominates** b. We may also write $b \lhd_C a$, saying that b **is dominated by** a. We say that a **weakly** dominates b, $a \unrhd b$, if only $(\forall c \in C)(aS_c b)$. Since we are working with true criteria we have that $a \unrhd_C b$ is equivalent with $\mathbf{a} \geq_{\mathcal{X}} \mathbf{b}$.

DOMINATES.
IS DOMINATED BY.
WEAK DOMINANCE RELATION.

4.5.3 Monotonicity

A basic principle[10] stemming from MCDA [97] is that $a \unrhd_C b \Rightarrow aSb$. On the other hand we have that $aSb \iff \lambda(a) \geq_{\mathcal{L}} \lambda(b)$. Merging all these expressions we find in a natural way the monotonicity constraint

$$\mathbf{a} \geq_{\mathcal{X}} \mathbf{b} \Rightarrow \lambda(a) \geq_{\mathcal{L}} \lambda(b)\,. \tag{4.5.1}$$

Since this constraint advocates $\mathbf{a} = \mathbf{b} \Rightarrow \lambda(a) = \lambda(b)$, it does not tolerate the presence of doubt, i.e. no uncertainty is allowed in the data, no errors. Thus, it is too restrictive for applications in supervised learning. The reason for this lies in the fact that we have adopted a principle from MCDA without considering its context: *build* a ranking based on the set C of criteria. This is a different setting than for supervised learning where we try to *reconstruct* a ranking based on the set C. In the former, the set C is a *framework*, in the latter, this same set C is a *restriction*.
We can solve this problem by applying the same principle but with the additional demand that we restrict our knowledge to the information we can retrieve from C. In that case, we define the dominance relation on $\Omega_{\mathcal{X}} \subseteq \mathcal{X}$, resulting in $\mathbf{x} \unrhd \mathbf{y}$ if and only if $\mathbf{x} \geq_{\mathcal{X}} \mathbf{y}$, and the principle becomes $\mathbf{x} \unrhd \mathbf{y} \Rightarrow \mathbf{x}\hat{S}\mathbf{y}$. Together with (4.4.3), this finally leads to the elementary monotonicity constraint

$$\mathbf{x} \geq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \unrhd_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})\,.$$

[10]For a more in depth treatise, see Section 6.2.3, p. 154.

In this case, doubt is tolerated since $\mathbf{x} = \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) = \lambda_{\text{repr}}(\mathbf{y})$ is a trivial demand. This also means that the **elementary monotonicity constraint** reduces to

ELEMENTARY MONOTONICITY CONSTRAINT.

$$\boxed{\mathbf{x} >_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglerighteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})} \tag{4.5.2}$$

We can now adapt the two equivalent definitions of Definition 4.5.1, which leads us to two different notions:

Definition 4.5.2

REVERSED PREFERENCE.

There are two different notions of **reversed preference**:

(i) There is **reversed preference** between the ranking $(\lambda, \geq_{\mathcal{L}})$ and the set of criteria C if

$$(\exists(a,b) \in \Omega^2)(\mathbf{a} >_{\mathcal{X}} \mathbf{b} \wedge \lambda(a) \not\geq_{\mathcal{L}} \lambda(b))\,.$$

(ii) There is **reversed preference** inside the representation $(\lambda_{\text{repr}}, \trianglerighteq_{\text{Im}})$ if

$$(\exists(\mathbf{x},\mathbf{y}) \in \Omega^2_{\mathcal{X}})(\mathbf{x} >_{\mathcal{X}} \mathbf{y} \wedge \lambda_{\text{repr}}(\mathbf{x}) \not\trianglerighteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y}))\,.$$

CONSISTENT.

A representation of a ranking is said to be **consistent** if there is no reversed preference inside it.

The two notions of reversed preference in Definition 4.5.2 do not coincide, we must make a clear distinction between the first definition that considers inconsistencies in the ranking, and the second definition that considers inconsistencies in a representation of a ranking.

About doubt and reversed preference. There is a major difference between the existence of reversed preference in a ranking, and the existence of doubt. People can accept doubt in a classification, but they will not accept reversed preference in a ranking. For example, you might accept that it is difficult to choose between two candidates, either because you feel you don't have enough information about them, or you feel they are too similar to differentiate, e.g. if athlete A wins the first 5 challenges in a decathlon, and ends second in the other 5, while athlete B ends second in the first 5, but wins the remaining 5. On the other hand, it is never tolerated that the candidate with the lower marks ends up in a higher rank than the candidate with the better marks, e.g. if athlete A always ends in the fourth or fifth place, while athlete B is always among the first three, then it is unacceptable that A would end up in a higher position on the league table than B.
This is nicely reflected in the definitions: there is never doubt between the classification $\hat{\lambda}$ and the corresponding set of attributes Q, but there can exist reversed preference between the ranking $(\hat{\lambda}, \leq^{[2]})$ and the corresponding set of criteria C. It is also the reason why we introduced the term *consistency* instead of *monotonicity*. While these two notions are clearly equivalent, "consistency" conveys the semantical idea behind the property of monotonicity of the representation.

Example 4.5.2. *Remember the pet we wanted to buy? Once arrived at the pet shop, we snooped a bit around, and came across an Anatolian Shepherd Dog puppy staring at us with his adorable puppy eyes, and you feel directly there is a bond between you and this little fellow .... However, this puppy was destined to become huge, not just big, but really huge! According to the shop owner, the puppy would grow to a calm dog with an even temperament of approximately 74cm at the withers and a good 60 kilos (certainly enough mass to qualify for a house demolition kind of huge). Still, he was so adorable.... there you go, reversed preference w.r.t. the criterion "size".*
You come home and tell your (girl/boy)friend/husbund/wife of this one puppy that made your heart melt. However, your friend did not see the puppy, and only sees the potential damage it can and will do in the future, once this adorable puppy starts reaching bigger proportions. He/she cannot accept your decision that this puppy is ranked higher than for example a Lakeland Terrier.

4.6 Transforming reversed preference into doubt

4.6.1 Introduction

As just mentioned, the occurrence of reversed preference in $\hat{\lambda}$ is not satisfactory (even unacceptable). This can be solved by redefining λ (or in terms of supervised learning, altering the training data) in such a way that the reversed preferences disappear. A very drastic solution could be to demand that $\mathbf{a} <_{\mathcal{X}} \mathbf{b} \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)$ for the training data. A less drastic one, is to find a maximally consistent subset by eliminating some of the data. Another possibility (see below) is to redefine C until the resulting $\hat{\lambda}$ behaves monotonically according to (4.5.2). All these proposals have an invasive character, and might even be unfeasible in certain circumstances. We therefore propose another, non-invasive method, that uses all available information, and results in the closest possible consistent representation by defining a mapping $\tilde{\lambda}$ such that $(\tilde{\lambda}, \leq^{[2]})$ does no longer contain reversed preferences.

4.6.2 Sources of reversed preference

We begin this section by an enumeration of how reversed preferences can arise in the ranking problem. In this case, the classification λ on Ω (or rather on a finite sample $\mathcal{S} \subseteq \Omega$) and the order $\leq_{\mathcal{L}}$ on $\mathcal{L}$ are furnished by a Decision Maker (DM). The DM also gives the set of criteria C to be considered. We first focus on reversed preference between the ranking $(\lambda, \leq_{\mathcal{L}})$ in Ω and the set of criteria C, i.e. there exist objects $a, b \in \mathcal{S}$ such that

$$\mathbf{a} <_{\mathcal{X}} \mathbf{b} \wedge \lambda(a) \not\leq_{\mathcal{L}} \lambda(b)\,.$$

Now assuming that reversed preference occurs, there are several scenarios possible for how these reversed preferences came into being:

(i) The DM has based his decisions on the set of criteria C:

   (a) The DM has used some additional information not present in C. A solution could be to find the criteria $c_1, \ldots, c_r \notin C$ the DM uses and add them to C.

   (b) The DM has made an inconsistent decision. In this case, the mapping λ might be redefined in a consistent way by the DM. If this is not a possibility (e.g. the DM has no time, the samples are taken from past decisions, $\ldots$), both C and λ must remain unchanged. We will show next how we can deal with this case.

   (c) It is agreed that C is the final set of criteria to be considered. Again we have to keep C and $(\lambda, \leq_{\mathcal{L}})$, including the conflict between them.

(ii) The DM has made his decisions before the set C was defined[11]:

   (a) Some meaningful criteria $c_1, \ldots, c_r \notin C$ were missed in the first attempt to construct the set C, such that $(\exists c \in \{c_1, \ldots, c_r\})(a \neg S_c b)$. This means that a does no longer dominate b w.r.t. $C \cup \{c_1, \ldots, c_r\}$.

   (b) It is agreed that C is the final set of criteria to be considered. We are in the same situation as case (i–c).

(iii) There is more than one DM involved, and they do not share the same preferences, and/or they used different information, and/or they made a mistake.

Example 4.6.1. *Ok, so the puppy with the big eyes caused reversed preference. This is clearly a case of (1a), there is some additional information not present in the your first set of criteria that only contained "size" and "non-nervousness". Once the decision had to be made, in your mind you also added the criteria "adorable" and "bonds with me".*

Since for practical purposes we use the representation of the ranking, it is not really necessary to solve the problems just stated. Indeed, we must only take care of reversed preference inside the representation $(\hat{\lambda}, \leq^{[2]})$ because of the following proposition:

Proposition 4.6.2. *Substituting the ranking $(\lambda, \leq_{\mathcal{L}})$ by its representation $(\hat{\lambda}, \leq^{[2]})$*

(i) might eliminate existing reversed preference (i.e. if $\mathbf{a} <_{\mathcal{X}} \mathbf{b}$ and $\lambda(a) \not\leq_{\mathcal{L}} \lambda(b)$, then it can happen that $\hat{\lambda}(\mathbf{a}) \leq^{[2]} \hat{\lambda}(\mathbf{b})$.)

(ii) will never introduce new reversed preferences.

Proof.
We will illustrate the first obvious assertion with a little example. Consider a ranking $(\lambda, \leq_{\mathcal{L}})$ with $\mathcal{L} = \{1, 2, 3\}$, $1 <_{\mathcal{L}} 2 <_{\mathcal{L}} 3$, $\{a, b, c, d\} \subseteq \Omega$ and $\mathbf{a} = \mathbf{b} = \mathbf{x} <_{\mathcal{X}}$

[11] This means that the DM could not have made an inconsistent decision, because there are no criteria yet to be inconsistent with.

$\mathbf{c} = \mathbf{d} = \mathbf{y}$. Furthermore, assume that $\lambda(a) = 1, \lambda(b) = 3, \lambda(c) = 2, \lambda(d) = 3$. This means b and c give rise to reversed preference. However, we have $\hat{\lambda}(\mathbf{x}) = [1, 3] \leq^{[2]} \hat{\lambda}(\mathbf{y}) = [2, 3]$ and thus $\mathbf{x}$ and $\mathbf{y}$ do not give rise to any reversed preferences.
To prove the second assertion, consider a ranking $(\lambda, \leq_{\mathcal{L}})$. For $a_1, a_2, b_1, b_2 \in \Omega$, we assume there is doubt: $\mathbf{a}_1 = \mathbf{a}_2 = \mathbf{x}$ and $\mathbf{b}_1 = \mathbf{b}_2 = \mathbf{y}$. We put $\lambda(a_1) = r_1, \lambda(a_2) = r_2, \lambda(b_1) = s_1, \lambda(b_2) = s_2$. Without restrictions we may assume that $r_1 \leq_{\mathcal{L}} r_2$, $s_1 \leq_{\mathcal{L}} s_2$, $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$, $\hat{\lambda}(\mathbf{x}) = [r_1, r_2]$ and $\hat{\lambda}(\mathbf{y}) = [s_1, s_2]$. Now suppose there is no reversed preference between $\mathbf{x}$ and $\mathbf{y}$ w.r.t. $(\lambda, \leq_{\mathcal{L}})$ and C, i.e. $(\forall a \in \Omega)(\mathbf{a} = \mathbf{x})$ and $(\forall b \in \Omega)(\mathbf{b} = \mathbf{y})$ we have $\lambda(a) \leq_{\mathcal{L}} \lambda(b)$. This means we must have that $r_2 \leq_{\mathcal{L}} s_1$, which automatically leads to $\hat{\lambda}(\mathbf{x}) = [r_1, r_2] \leq^{[2]} \hat{\lambda}(\mathbf{y}) = [s_1, s_2]$. □

4.6.3 Dealing with reversed preference: intervals

The above proposition implies that if there is reversed preference inside the representation $(\hat{\lambda}, \leq^{[2]})$, it must have its origins in one of the situations described above. If both λ and C should remain unchanged, as in cases (i–b), (i–c) and (ii–b), we may transform $\hat{\lambda}$ into a mapping $\tilde{\lambda}$ such that $(\tilde{\lambda}, \leq^{[2]})$ does no longer contain reversed preference. This transformation should stay as close as possible to the original $\hat{\lambda}$, and if there is no reversed preference in $(\hat{\lambda}, \leq^{[2]})$, then $\tilde{\lambda}$ should be equal to $\hat{\lambda}$.
We will now show how this transformation can be done. Assume there is reversed preference inside the representation $(\hat{\lambda}, \leq^{[2]})$, meaning we can find $\mathbf{x}, \mathbf{y} \in \Omega_{\mathcal{X}}$ such that

$$\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \text{ and } \hat{\lambda}(\mathbf{x}) = [r_1, r_2] \not\leq^{[2]} \hat{\lambda}(\mathbf{y}) = [s_1, s_2] \,.$$

Because of expression (4.4.2), we can never make the interval $[r_1, r_2]$ or $[s_1, s_2]$ smaller without removing an object from the sample space. This would result in neglecting the information that does not fit our formalisation. Thus, removing objects cannot be defended. As a consequence, we may only enlarge the intervals in order to remove the inconsistency. To stay as close as possible to the original (inconsistent) information, we will enlarge the intervals in a minimal way to eliminate the reversed preference. We have

$$[r_1, r_2] \not\leq^{[2]} [s_1, s_2] \iff (r_1 >_{\mathcal{L}} s_1) \vee (r_2 >_{\mathcal{L}} s_2) \,,$$

and we must try to find intervals $[\tilde{r}_1, \tilde{r}_2] \supseteq [r_1, r_2]$ and $[\tilde{s}_1, \tilde{s}_2] \supseteq [s_1, s_2]$ such that $[\tilde{r}_1, \tilde{r}_2] \leq^{[2]} [\tilde{s}_1, \tilde{s}_2]$ or still $(\tilde{r}_1 \leq_{\mathcal{L}} \tilde{s}_1) \wedge (\tilde{r}_2 \leq_{\mathcal{L}} \tilde{s}_2)$.

(i) $s_1 <_{\mathcal{L}} r_1$. We may only resolve this by reducing s_1 to some $\tilde{s}_1$ and r_1 to some $\tilde{r}_1$. It is obvious that we can turn the inequality into an equality if we put $\tilde{r}_1 := s_1$ and keep $\tilde{s}_1 := s_1$. Moreover, this is the smallest[12] change possible in order to obtain $\tilde{r}_1 \leq_{\mathcal{L}} \tilde{s}_1$ with $\tilde{r}_1 \leq_{\mathcal{L}} r_1$ and $\tilde{s}_1 \leq_{\mathcal{L}} s_1$.

[12]There is only one acceptable distance measure on an ordinal scale, namely the distance in the underlying graph.

(ii) $s_2 <_{\mathcal{L}} r_2$. The smallest possible change to obtain $\tilde{r}_2 \leq_{\mathcal{L}} \tilde{s}_2$ with $\tilde{r}_2 \geq_{\mathcal{L}} r_2$ and $\tilde{s}_2 \geq_{\mathcal{L}} s_2$, is putting $\tilde{r_2} := r_2$ and $\tilde{s_2} := r_2$.

So, we may eliminate this reversed preference by subjecting $\hat{\lambda}$ to the transformation t:

$$\begin{aligned} t(\hat{\lambda}(\mathbf{x})) &= t([\hat{\lambda}_\ell(\mathbf{x}), \hat{\lambda}_r(\mathbf{x})]) &:=& [\min(\hat{\lambda}_\ell(\mathbf{x}), \hat{\lambda}_\ell(\mathbf{y})), \hat{\lambda}_r(\mathbf{x})]\,, \\ t(\hat{\lambda}(\mathbf{y})) &= t([\hat{\lambda}_\ell(\mathbf{y}), \hat{\lambda}_r(\mathbf{y})]) &:=& [\hat{\lambda}_\ell(\mathbf{y}), \max(\hat{\lambda}_r(\mathbf{x}), \hat{\lambda}_r(\mathbf{y}))]\,. \end{aligned}$$

Moreover, if there was no reversed preference, this transformation would just yield

$$t(\hat{\lambda}(\mathbf{x})) = \hat{\lambda}(\mathbf{x}) \quad \text{and} \quad t(\hat{\lambda}(\mathbf{y})) = \hat{\lambda}(\mathbf{y})\,.$$

It should be noted that this transformation may create new reversed preferences. The previous procedure must therefore be repeated until no more reversed preferences exist. It is clear that in the present finite setting, there is convergence, in the worst case we would end up with the constant mapping to $[\inf \mathcal{L}, \sup \mathcal{L}]$.

Theorem 4.6.3. *Let $(\lambda, \leq_{\mathcal{L}})$ be a ranking in Ω, and let C be a set of criteria. Now define*

$$\boxed{\begin{array}{cccl} \tilde{\lambda} & : & \Omega_{\mathcal{X}} & \to & \mathcal{L}^{[2]}, \\ & & \mathbf{x} & \mapsto & [\min_{\mathbf{y}\in[\mathbf{x})} \hat{\lambda}_\ell(\mathbf{y}), \max_{\mathbf{y}\in(\mathbf{x}]} \hat{\lambda}_r(\mathbf{y})]\,, \end{array}}$$

where $[\mathbf{x}) = \{\mathbf{x}' \in \Omega_{\mathcal{X}} \mid \mathbf{x} \leq_{\mathcal{X}} \mathbf{x}'\}$ and $(\mathbf{x}] = \{\mathbf{x}' \in \Omega_{\mathcal{X}} \mid \mathbf{x}' \leq_{\mathcal{X}} \mathbf{x}\}$. We have that

(i) *The representation $(\tilde{\lambda}, \leq^{[2]})$ is consistent with C.*

(ii) *For all $\mathbf{x} \in \mathcal{X}$, it holds that $\hat{\lambda}(\mathbf{x}) \subseteq \tilde{\lambda}(\mathbf{x})$, and if $(\hat{\lambda}, \leq^{[2]})$ is consistent, then $\tilde{\lambda} = \hat{\lambda}$.*

(iii) *There exists no other representation $(\overline{\lambda}, \leq^{[2]})$ consistent with C such that for all $\mathbf{x} \in \mathcal{X}$, it holds that $\hat{\lambda}(\mathbf{x}) \subseteq \overline{\lambda}(\mathbf{x}) \subseteq \tilde{\lambda}(\mathbf{x})$.*

Proof.
(i) First note that $\min_{\mathbf{y}\in[\mathbf{x})} \hat{\lambda}_\ell(\mathbf{y}) \leq_{\mathcal{L}} \max_{\mathbf{y}\in(\mathbf{x}]} \hat{\lambda}_r(\mathbf{y})$ since $[\mathbf{x}) \cap (\mathbf{x}] = \{\mathbf{x}\}$. Now consider $\mathbf{x}, \mathbf{y} \in \Omega_{\mathcal{X}}$ such that $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$. This implies that $\mathbf{x} \in (\mathbf{y}]$ and $\mathbf{y} \in [\mathbf{x})$, and so $(\mathbf{x}] \subseteq (\mathbf{y}]$ and $[\mathbf{y}) \subseteq [\mathbf{x})$ (see Figure 4.6). From this it immediately follows that

$$\tilde{\lambda}(\mathbf{x}) = [\min_{\mathbf{z}\in[\mathbf{x})} \hat{\lambda}_\ell(\mathbf{z}), \max_{\mathbf{z}\in(\mathbf{x}]} \hat{\lambda}_r(\mathbf{z})] \leq^{[2]} \tilde{\lambda}(\mathbf{y}) = [\min_{\mathbf{z}\in[\mathbf{y})} \hat{\lambda}_\ell(\mathbf{z}), \max_{\mathbf{z}\in(\mathbf{y}]} \hat{\lambda}_r(\mathbf{z})]\,.$$

(ii) Self-evident.
(iii) Let $\overline{\lambda}(\neq \tilde{\lambda})$ be such that for all $\mathbf{x} \in \mathcal{X}$, it holds that $\hat{\lambda}(\mathbf{x}) \subseteq \overline{\lambda}(\mathbf{x}) \subseteq \tilde{\lambda}(\mathbf{x})$. Remark that, because of (ii), this is only possible if $(\hat{\lambda}, \leq^{[2]})$ is not consistent. We have that there exists at least one $\mathbf{x} \in \mathcal{X}$ such that $\overline{\lambda}(\mathbf{x}) = [l, r] \subset \tilde{\lambda}(\mathbf{x})$. Assume that $l > \min_{\mathbf{y}\in[\mathbf{x})} \hat{\lambda}_\ell(\mathbf{y})$. Evidently, this implies there exist a non-empty set Y

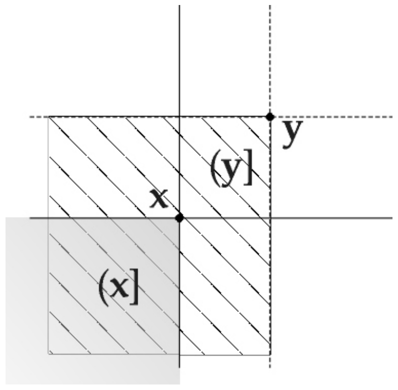


Figure 4.6: $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$

of $\mathbf{y} >_{\mathcal{X}} \mathbf{x}$ such that $\hat{\lambda}_\ell(\mathbf{y}) = i <_{\mathcal{L}} \hat{\lambda}_\ell(\mathbf{x}) = j$. Let $\mathbf{y}^* \in \arg\min_{\mathbf{y}\in Y} \hat{\lambda}_\ell(\mathbf{y})$, and $i^* = \hat{\lambda}_\ell(\mathbf{y}^*)$. Since $(\forall \mathbf{z} \in \mathcal{X})(\hat{\lambda}(\mathbf{z}) \subseteq \overline{\lambda}(\mathbf{z}))$, we have that $j \in \overline{\lambda}(\mathbf{x})$ and $i^* \in \overline{\lambda}(\mathbf{y}^*)$. If $(\hat{\lambda}, \leq^{[2]})$ is to be consistent, it must hold that $i^* \in \overline{\lambda}(\mathbf{x})$. By construction, we have that $i^* = \min_{\mathbf{y}\in[\mathbf{x})} \hat{\lambda}_\ell(\mathbf{y}) < l$, a contradiction. □

Essentially, we have enlarged the intervals in a minimal way such that there are no more violations against the monotonicity requirement (4.5.2). In other words, we transform the unacceptable reversed preferences into acceptable doubt.

Example 4.6.4. *Let us demonstrate this on a small example taken from [48]. Assume we have $\Omega = \{a_1, \dots, a_6\}$, a single criterion $c : \Omega \to (\{1, \dots, 6\}, \leq)$ and a ranking $(\lambda, \leq)$ with $f : \Omega \to \{1, \dots 6\}$, as shown in Table 4.1 (see also p. 69). There is no doubt, so $\Omega \cong \Omega_{\mathcal{X}}$. Table 4.2 lists the consistent representation of this ranking.*

	a_1	a_2	a_3	a_4	a_5	a_6
c	2	1	4	3	6	5
λ	1	2	3	4	5	6

Table 4.1: A simple ranking $(\lambda, \leq)$.

	$\mathbf{a}_2$	$\leq_{\mathcal{X}}$	$\mathbf{a}_1$	$\leq_{\mathcal{X}}$	$\mathbf{a}_4$	$\leq_{\mathcal{X}}$	$\mathbf{a}_3$	$\leq_{\mathcal{X}}$	$\mathbf{a}_6$	$\leq_{\mathcal{X}}$	$\mathbf{a}_5$
$\hat{\lambda}$	2		1		4		3		6		5
$\tilde{\lambda}$	$[1,2]$	$\leq^{[2]}$	$[1,2]$	$\leq^{[2]}$	$[3,4]$	$\leq^{[2]}$	$[3,4]$	$\leq^{[2]}$	$[5,6]$	$\leq^{[2]}$	$[5,6]$

Table 4.2: The consistent representation $(\tilde{\lambda}, \leq^{[2]})$ of $(\lambda, \leq)$.

4.7 Representing a ranking: distributions

4.7.1 Introduction

Up to now, we have focussed on classifiers (functions with domain $\Omega_{\mathcal{X}}$) that return the possible values that can be assigned to each object. In this section, we go one step further and consider probabilistic classifiers $\hat{\lambda}_{\text{prob}}$ assigning to each element of $\Omega_{\mathcal{X}}$ a probability distribution over the labels. (In case Ω is finite, this is easily achieved by normalising the frequency distributions associated with the elements of $\Omega_{\mathcal{X}}$.) As stated in the framework of Chapter 2, we shift our view from the *set interpretation* to the *distribution interpretation* (see p. 29).

4.7.2 Stochastic dominance

$\mathcal{F}(\mathcal{L})$ As before, we need to establish an order on the set $\mathcal{F}(\mathcal{L})$ of all possible probability distributions[13] over $\mathcal{L}$. In the present context, the ordering that comes immediately to mind is the stochastic dominance ordering. Let $f_X, f_Y \in \mathcal{F}(\mathcal{L})$, and denote their cumulative distribution functions as F_X and F_Y, i.e. $F_X(i) = \mathcal{P}(X \leq i)$. **Weak (first order) stochastic dominance** $\trianglerighteq_{(1)}$ is defined by

STOCHASTIC DOMINANCE.

$$f_X \trianglerighteq_{(1)} f_Y \iff (\forall i \in \mathcal{L})(F_X(i) \leq F_Y(i)) \,.$$

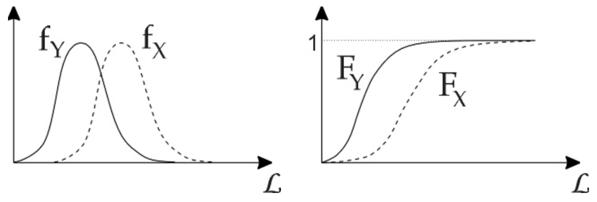


Figure 4.7: Stochastic dominance (continuous case).

Remark that if we consider the support $= \{\ell \in \mathcal{L} \mid f(\ell) > 0\}$, then

$$f_X \trianglerighteq_{(1)} f_Y$$
$$\Downarrow$$
$$[\min, \max] \geq^{[2]} [\min, \max] \,.$$

Of course, the converse does not hold, e.g. for $\mathcal{L} = \{1,2,3\}$, $f_X = (0.4, 0.4, 0.2)$ and $f_Y = (0.2, 0.8, 0)$.
Along the same line, we have the following lemma:

[13] If $\mathcal{L} = \{1, \ldots, k\}$, then we denote a distribution $f_X \in \mathcal{F}(\mathcal{L})$ as a vector of dimension k: $f_X = (\mathcal{P}(X = 1), \ldots, \mathcal{P}(X = k))$.

Lemma 4.7.1. *If we identify each interval of $\mathcal{L}^{[2]}$ with a uniform distribution functions over this interval, the order $\leq^{[2]}$ just coincides with $\trianglelefteq_{(1)}$.*

Proof.
This can be easily seen in Figure 4.8, where we consider 2 intervals I_1 and I_2, and the corresponding the cumulative uniform distributions functions F_i over the intervals I_i (for $i = 1, 2$). The other configurations not depicted in the figure are equally obvious.

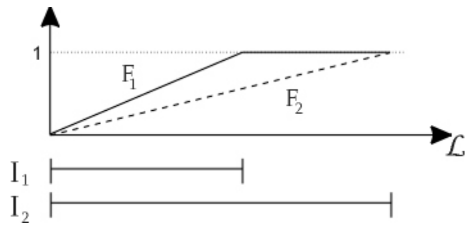


Figure 4.8: Intervals and uniform distributions.

4.7.3 Meaningful representations

For a representation to be meaningful, it should at least make the second assertion of Proposition 4.6.2 true. Because of its importance, we will look for a class of representations that can guarantee this assertion. This can easily be done by imposing a kind of minimal consistency on the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$.

Definition 4.7.1

We say that the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ of $(\lambda, \leq_{\mathcal{L}})$ is **minimally consistent** in $U \subseteq \mathcal{X}$, if for all $\mathbf{x}, \mathbf{y} \in U$ with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$, it holds that

MINIMALLY CONSISTENT.

$$\hat{\lambda}_r(\mathbf{x}) \leq_{\mathcal{L}} \hat{\lambda}_\ell(\mathbf{y}) \implies \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y}) .$$

Obviously, any consistent representation is also minimally consistent. Similar to Proposition 4.6.2 we can prove that

Proposition 4.7.2. *Substituting the ranking $(\lambda, \leq_{\mathcal{L}})$ by a representation $(\lambda_{repr}, \trianglelefteq_{Im})$ that is minimally consistent in $\Omega_{\mathcal{X}}$ will never introduce new reversed preferences.*

It is clear that $(\hat{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ satisfies Definition 4.7.1. Moreover, also the first part of Proposition 4.6.2 can be kept using the same proof, but this cannot be generalised for all $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$.

4.7.4 Consistent representations

In this new setting, we aim once more at finding a representation without reversed preferences, and preferably without reducing the support of the distributions (which is equivalent to not removing any objects as advocated in Section 4.6). Consider an $\mathbf{x} \in \Omega_{\mathcal{X}}$, with $\hat{\lambda}_{\text{prob}}(\mathbf{x}) = f_{\mathbf{x}}$. We know that for all $\mathbf{y} \leq_{\mathcal{X}} \mathbf{x}$ (and of course $\mathbf{y} \in \Omega_{\mathcal{X}}$), it should hold that $f_{\mathbf{y}} \trianglelefteq_{(1)} f_{\mathbf{x}}$ or $F_{\mathbf{y}}(i) \geq F_{\mathbf{x}}(i)$, and therefore

$$F_{\mathbf{x}}(i) \leq \min_{\mathbf{y} \in (\mathbf{x}]} F_{\mathbf{y}}(i)\,.$$

At the same time, for all $\mathbf{y} \geq_{\mathcal{X}} \mathbf{x}$, it should hold that $f_{\mathbf{x}} \trianglelefteq_{(1)} f_{\mathbf{y}}$, or

$$F_{\mathbf{x}}(i) \geq \max_{\mathbf{y} \in [\mathbf{x})} F_{\mathbf{y}}(i)\,.$$

However, it may very well happen (if $(\hat{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ contains reversed preferences) that $\min_{\mathbf{y} \in (\mathbf{x}]} F_{\mathbf{y}}(i) < \max_{\mathbf{y} \in [\mathbf{x})} F_{\mathbf{y}}(i)$. For example, with $\Omega_{\mathcal{X}} = \{\mathbf{x}, \mathbf{y}\}$, $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$, $\mathcal{L} = \{1, 2\}$, $f_{\mathbf{x}} = (0, 1)$ and $f_{\mathbf{y}} = (1, 0)$, we find the constraints $F_{\mathbf{x}}(1) \leq 0$ and $F_{\mathbf{x}}(1) \geq 1$.

Theorem 4.7.3. *Let $(\lambda, \leq_{\mathcal{L}})$ be a ranking on Ω. For all $\mathbf{y} \in \Omega_{\mathcal{X}}$, we denote $\hat{\lambda}_{\text{prob}}(\mathbf{y}) = f_{\mathbf{y}}$. Let $s \in [0, 1]$. For all $\mathbf{x} \in \Omega_{\mathcal{X}}$, for all $i \in \mathcal{L}$, we define:*

$$\boxed{\tilde{F}_{\mathbf{x}}(i) := (1 - s) \cdot F_m(\mathbf{x}, i) + s \cdot F_M(\mathbf{x}, i)}$$

where

$$\begin{aligned} F_m(\mathbf{x}, i) &= \min_{\mathbf{y} \in (\mathbf{x}]} F_{\mathbf{y}}(i)\,, \qquad \textit{with} \quad (\mathbf{x}] = \{\mathbf{x}' \in \Omega_{\mathcal{X}} \mid \mathbf{x}' \leq_{\mathcal{X}} \mathbf{x}\}\,, \\ F_M(\mathbf{x}, i) &= \max_{\mathbf{y} \in [\mathbf{x})} F_{\mathbf{y}}(i)\,, \qquad \textit{with} \quad [\mathbf{x}) = \{\mathbf{x}' \in \Omega_{\mathcal{X}} \mid \mathbf{x} \leq_{\mathcal{X}} \mathbf{x}'\}\,. \end{aligned}$$

If we set $\tilde{\lambda}_{\text{prob}}(\mathbf{x}) = \tilde{f}_{\mathbf{x}}$, then we have that

(i) *$(\tilde{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ is consistent.*

(ii) *If $(\hat{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ is consistent, then $\tilde{f}_{\mathbf{x}} = \hat{\lambda}_{\text{prob}}(\mathbf{x})$, for all $\mathbf{x} \in \Omega_{\mathcal{X}}$.*

Proof.
Firstly, remark that $\tilde{F}_{\mathbf{x}}$ may indeed be regarded as a cumulative distribution function. Indeed, since $i \leq_{\mathcal{L}} j \Rightarrow (\forall \mathbf{y} \in \Omega_{\mathcal{X}})(F_{\mathbf{y}}(i) \leq F_{\mathbf{y}}(j))$, we find that $F_m(\mathbf{x}, \cdot)$ and $F_M(\mathbf{x}, \cdot)$ are non-decreasing (in their second argument), and hence also $\tilde{F}_{\mathbf{x}}$. Clearly $\tilde{F}_{\mathbf{x}}(\max \mathcal{L}) = 1$.

(i) Now consider $\mathbf{x}, \mathbf{y} \in \Omega_{\mathcal{X}}$, with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$. We need to prove that $\tilde{f}_{\mathbf{x}} \trianglelefteq_{(1)} \tilde{f}_{\mathbf{y}}$. We know that $(\mathbf{x}] \subseteq (\mathbf{y}]$ and $[\mathbf{x}) \supseteq [\mathbf{y})$. Let $i \in \mathcal{L}$. We have

$$\begin{aligned} \min_{\mathbf{z} \in (\mathbf{x}]} F_{\mathbf{z}}(i) &\geq \min_{\mathbf{z} \in (\mathbf{y}]} F_{\mathbf{z}}(i)\,, \\ \max_{\mathbf{z} \in [\mathbf{x})} F_{\mathbf{z}}(i) &\geq \max_{\mathbf{z} \in [\mathbf{y})} F_{\mathbf{z}}(i)\,, \end{aligned}$$

i.e. $F_m(\cdot, i)$ and $F_M(\cdot, i)$ are non-increasing (in their first argument). Hence $\tilde{F}_{\mathbf{x}}(i) \geq \tilde{F}_{\mathbf{y}}(i)$.

(ii) When $(\hat{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ is consistent, we have that $F_m(\mathbf{x}, i) = \min_{\mathbf{y}\in(\mathbf{x}]} F_{\mathbf{y}}(i) = F_{\mathbf{x}}(i)$ and $F_M(\mathbf{x}, i) = \max_{\mathbf{y}\in[\mathbf{x})} F_{\mathbf{y}}(i) = F_{\mathbf{x}}(i)$. □

The proposed family of solutions to the problem is certainly not the sole one. This in severe contrast with the unique solution put forward in Theorem 4.6.3. Also notice that the support is not reduced for $s \in]0, 1[$.

Example 4.7.4. *A small example demonstrating the above proposition is given in Table 4.3.*

	$\mathbf{x}_1$	$\leq_{\mathcal{X}}$	$\mathbf{x}_2$	$\leq_{\mathcal{X}}$	$\mathbf{x}_3$
$\hat{\lambda}_{\text{prob}}(\mathbf{x}) = f_{\mathbf{x}}$	$(0.4, 0.4, 0.2)$		$(0.2, 0.8, 0.0)$		$(0.2, 0.3, 0.5)$
$F_{\mathbf{x}}$	$(0.4, 0.8, 1.0)$		$(0.2, 1.0, 1.0)$		$(0.2, 0.5, 1.0)$
$\tilde{F}_{\mathbf{x}}$	$(0.4, 0.9, 1.0)$		$(0.2, 0.9, 1.0)$		$(0.2, 0.5, 1.0)$
$\tilde{\lambda}_{\text{prob}}(x)$	$(0.4, 0.5, 0.1)$	$\trianglelefteq_{(1)}$	$(0.2, 0.7, 0.1)$	$\trianglelefteq_{(1)}$	$(0.2, 0.3, 0.5)$

Table 4.3: The consistent representation $(\tilde{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$ of $(\lambda, \leq)$ using $s = \frac{1}{2}$.

We could avoid introducing the parameter s by simply defining an interval function, as we did in Section 4.6.3. So instead of pint-pointing a specific probability distribution for each object, we only designate an indicative region wherein the probability distributions can be found. This approach is more honest in that it does not give a false impression of accurateness in the case there is in fact not enough information available to be accurate.
We now define the notion of an interval of (cumulative) distributions:

Definition 4.7.2

Let f_X and f_Y be two probability distributions over $\mathcal{L}$, with $f_X \trianglelefteq_{(1)} f_Y$. Then we define a **(probability distribution) interval** as

(PROBABILITY DISTRIBUTION) INTERVAL $[f_X, f_Y]$.

$$[f_X, f_Y] = \{f_Z \in \mathcal{F}(\mathcal{L}) \mid f_X \trianglelefteq_{(1)} f_Z \trianglelefteq_{(1)} f_Y\}\,.$$

We choose to denote the corresponding interval of cumulative distributions as

$$[F_X, F_Y] = \{F_Z \in \mathcal{F}_{\text{cum}}(\mathcal{L}) \mid (\forall i \in \mathcal{L})(F_X(i) \geq F_Z(i) \geq F_Y(i))\}\,,$$

where $\mathcal{F}_{\text{cum}}(\mathcal{L})$ is the set of cumulative distribution functions F corresponding to an $f \in \mathcal{F}(\mathcal{L})$.

This approach leads to the following non-invasive definition of $\tilde{F}_{\mathbf{x}}$ as an interval:

$$\tilde{F}_{\mathbf{x}} := [\tilde{F}^{\ell}_{\mathbf{x}}, \tilde{F}^{r}_{\mathbf{x}}]\,,$$

where

$$\begin{aligned}\tilde{F}^{\ell}_{\mathbf{x}}(i) &= \max(F_m(\mathbf{x},i), F_M(\mathbf{x},i))\,,\\ \tilde{F}^{r}_{\mathbf{x}}(i) &= \min(F_m(\mathbf{x},i), F_M(\mathbf{x},i))\,,\end{aligned}$$

with

$$\begin{aligned}F_m(\mathbf{x},i) &= \min_{\mathbf{y}\in(\mathbf{x}]} F_{\mathbf{y}}(i)\,,\\ F_M(\mathbf{x},i) &= \max_{\mathbf{y}\in[\mathbf{x})} F_{\mathbf{y}}(i)\,.\end{aligned}$$

The function $\tilde{F}^{\ell}_{\mathbf{x}}$ can be interpreted as the cumulative distribution function of $\tilde{f}^{\ell}_{\mathbf{x}}$, and $\tilde{F}^{r}_{\mathbf{x}}$ as the cumulative distribution function of $\tilde{f}^{r}_{\mathbf{x}}$. We obviously have that $\tilde{F}^{\ell}_{\mathbf{x}}(i) \geq \tilde{F}^{r}_{\mathbf{x}}(i)$, whence $\tilde{f}^{\ell}_{\mathbf{x}} \trianglelefteq_{(1)} \tilde{f}^{r}_{\mathbf{x}}$.

Proposition 4.7.5. *Let $(\lambda, \leq_{\mathcal{L}})$ be a ranking on Ω. Define the function*

$$\tilde{\lambda}_{prob}(\mathbf{x}) := [\tilde{f}^{\ell}_{\mathbf{x}}, \tilde{f}^{r}_{\mathbf{x}}]\,,$$

and the relation $\trianglelefteq^{[2]}_{(1)}$ as

$$[f_X, f_Y] \trianglelefteq^{[2]}_{(1)} [f_S, f_T] \iff (f_X \trianglelefteq_{(1)} f_S) \wedge (f_Y \trianglelefteq_{(1)} f_T)\,.$$

It holds that the representation $(\tilde{\lambda}_{prob}, \trianglelefteq^{[2]}_{(1)})$ is consistent.

As an alternative to Theorem 4.7.3, we now have:

Corollary 4.7.6. *If we define*

$$\tilde{\lambda}_{prob}(\mathbf{x}) = (1-s)\cdot \tilde{f}^{\ell}_{\mathbf{x}} + s\cdot \tilde{f}^{r}_{\mathbf{x}}\,,$$

then $(\tilde{\lambda}_{prob}, \trianglelefteq_{(1)})$ is consistent.

4.8 Summary

GENERAL:
A (complete) ranking $(\lambda, \leq_{\mathcal{L}})$ on Ω consists of a classification $f : \Omega \to \mathcal{L}$ and a (complete) order $\leq_{\mathcal{L}}$ on $\mathcal{L}$.

When the objects are described by a set of criteria C, Ω is represented by $\Omega_{\mathcal{X}} \subseteq \mathcal{X}$ and the representation of a ranking is in general written as:

$$(\lambda_{\text{repr}}, \trianglerighteq_{\text{Im}})\,.$$

The description by criteria results in the elementary monotonicity constraint

$$\mathbf{x} >_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglerighteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})\,.$$

<u>MONOTONE REPRESENTATION</u>:
We describe 3 different monotone representations $(\lambda_{\text{repr}}, \trianglerighteq_{\text{Im}})$:

- <u>set-based</u>: (based on $\hat{\lambda}(\mathbf{x}) = \{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}$)

  $\boxed{1}$ $(\tilde{\lambda}, \leq^{[2]})$, with $\tilde{\lambda} : \Omega_{\mathcal{X}} \to \mathcal{L}^{[2]}$,
  $$\mathbf{x} \mapsto [\min_{\mathbf{y} \in [\mathbf{x})} \hat{\lambda}_\ell(\mathbf{y}), \max_{\mathbf{y} \in (\mathbf{x}]} \hat{\lambda}_r(\mathbf{y})]\,,$$

  $$\hat{\lambda}_\ell(\mathbf{x}) = \min\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}\,,$$
  $$\hat{\lambda}_r(\mathbf{x}) = \max\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} = \mathbf{x}\}\,,$$

  and $\leq^{[2]}$ the order on $\mathcal{L}^{[2]}$ defined by

  $$[r_1, r_2] \leq^{[2]} [s_1, s_2] \iff (r_1 \leq_{\mathcal{L}} s_1) \wedge (r_2 \leq_{\mathcal{L}} s_2)\,.$$

- <u>distribution-based</u>: (based on $\hat{\lambda}_{\text{prob}}(\mathbf{x}) = f_{\mathbf{x}}$, the probability distribution of $\mathbf{x}$ over $\mathcal{L}$ according to λ)

  $\boxed{2}$ $(\tilde{\lambda}_{\text{prob}}, \trianglelefteq_{(1)})$, with $\tilde{\lambda}_{\text{prob}} : \Omega_{\mathcal{X}} \to \mathcal{F}(\mathcal{L})$,
  $$\mathbf{x} \mapsto \tilde{f}_{\mathbf{x}}\,,$$

  $$\tilde{F}_{\mathbf{x}}(i) = (1-s) \cdot F_m(\mathbf{x}, i) + s \cdot F_M(\mathbf{x}, i)\,,$$
  $$F_m(\mathbf{x}, i) = \min_{\mathbf{y} \in (\mathbf{x}]} F_{\mathbf{y}}(i)\,, \qquad F_M(\mathbf{x}, i) = \max_{\mathbf{y} \in [\mathbf{x})} F_{\mathbf{y}}(i)\,,$$

  and $\trianglelefteq_{(1)}$ the weak first order stochastic dominance relation.

  $\boxed{3}$ $(\tilde{\lambda}_{\text{prob}}, \trianglelefteq_{(1)}^{[2]})$, with $\tilde{\lambda}_{\text{prob}} : \Omega_{\mathcal{X}} \to \mathcal{F}(\mathcal{L})^{[2]}$,
  $$\mathbf{x} \mapsto [\tilde{f}^\ell_{\mathbf{x}}, \tilde{f}^r_{\mathbf{x}}]\,,$$

  $$\tilde{F}^\ell_{\mathbf{x}}(i) = \max(F_m(\mathbf{x}, i), F_M(\mathbf{x}, i))\,,$$
  $$\tilde{F}^r_{\mathbf{x}}(i) = \min(F_m(\mathbf{x}, i), F_M(\mathbf{x}, i))\,,$$

  and $\trianglelefteq_{(1)}^{[2]}$ the order on $\mathcal{F}(\mathcal{L})^{[2]}$ defined by

  $$[f_X, f_Y] \trianglelefteq_{(1)}^{[2]} [f_S, f_T] \iff (f_X \trianglelefteq_{(1)} f_S) \wedge (f_Y \trianglelefteq_{(1)} f_T)\,.$$

Interlude

CREATIVE PROCESSES

Sometimes, my mind can be as predictably uncontrollable as the regular ebb and flow of the tides. Having tremendous amounts of work is one of the catalysts to such behaviour. Typically, when time runs the shortest, a frenzy of artistic creativity will surge and dominate my thoughts. Perfect testimonies are these interludes, and the piano that doesn't stop beckoning me (like just an instant ago, when it lured me into Chopin's magnificent Revolution Étude).
The other way round, when I want to take some time off, my mind keeps equally rushing and whirling with thoughts and ideas concerning my research.

MONTPELLIER. *Somewhere in February, I had planned a short trip to Montpellier (France), to visit a girlfriend and indulge in some rock climbing. During the last minutes preceding my departure, I made the mistake of running the OSDL algorithm on some newly obtained data sets, and hence, the last image burned on my retina before closing my door was a screen displaying some very good results on one, and some slightly disappointing results on another data set. And of course, during the whole journey, it was the "slightly disappointing" that kept rumbling in my head on the sedate rhythm of the thundering train. I soon discovered the heart of the problem, and the first clues towards its solution began dawning on me.*

A STROLL AT THE BEACH. *The day after I arrived, I had a nice solitary walk on the beach and let the wind blow away my continuous pondering. However, instead of a clear mind, I found a patch of sand on which I started to etch what became the essence of the Balanced OSDL algorithm. When I was finished, the clouds in my mind had dissipated completely, and I was finally free to enjoy the remainder of my stay in the south of France.*

Chapter 5

OSDL: Ordinal Stochastic Dominance Learner

5.1 Introduction

With the mortar and tools prepared in the previous chapter, the path has been paved to deal in a mathematically and semantically sound way with rankings in the context of supervised learning. In Chapter 4, we have given a definition, a representation when dealing with attributes (in which case we are on the territory of ordinal regression), and a consistent interval and stochastic representation when dealing with criteria.
Although our main interest was in trying to find acceptable representations of a ranking that could be presented to a decision maker, it appears that the resulting propositions also constitute a solid basis for an instance-based learning method, i.e. a learning method that stores the given learning samples (instances) into memory in some kind of format, and is able to deduce from them the class labels of unseen objects by some usually local extrapolation technique. As such, we provide an alternative to OLM, the Ordinal Learning Model [9]. That is the topic of the present chapter.

5.2 Supervised learning of a ranking

Notions and conventions. A function $f : (X, \leq_X) \to (Y, \leq_Y)$ between two *posets* (see p. 52) (sets equipped with an *order*, i.e. a reflexive, antisymmetric, transitive relation) is called **monotone** if for all $x, y \in X$ it holds that

MONOTONE FUNCTION.

$$x \leq_X y \Rightarrow f(x) \leq_Y f(y)\,.$$

CHAIN.

A **chain** is a completely ordered set $(X, \leq_X)$, i.e. for all $x, y \in X$ we have either $x \leq_X y$ or $y \leq_X x$.
Let f_X and f_Y be two probability distributions over a finite set $\mathcal{L}$, and denote their cumulative distribution functions as F_X and F_Y, i.e. $F_X(i) = \mathcal{P}(X \leq i)$, then the **weak (first order) stochastic dominance relation** (see p. 106) $\trianglelefteq_{(1)}$ is defined by

STOCHASTIC DOMINANCE.

$$f_X \trianglelefteq_{(1)} f_Y \iff (\forall i \in \mathcal{L})(F_X(i) \geq F_Y(i))\,.$$

The cardinality of a set X is denoted by $|X|$.

5.2.1 The classification problem

The basic problem. For some object space Ω, the goal is to attach labels from a finite set $\mathcal{L}$ to the objects in Ω, i.e. to find a classification $\lambda : \Omega \to \mathcal{L}$. This classification must be such that, for a given finite **learning sample** (also called **data set**) $\Lambda = (\mathcal{S}, d)$, where $\mathcal{S} \subseteq \Omega$ and $d : \mathcal{S} \to \mathcal{L}$, some **risk functional** (also called **error function**) $R(\lambda)$ is minimised. The functional R is typically based on a predefined **loss function** $\ell : \mathcal{L} \times \mathcal{L} \to \mathbb{R}$. For example the expected value of the losses on some test sample: $R = E[\ell(\lambda(a), d(a)) \mid a \in \mathcal{S}_{\text{test}}]$. The classification error is then obtained by choosing the loss function $\ell(i, j) = 1$ if $i = j$, and 0

LEARNING SAMPLE Λ. DATA SET. RISK FUNCTIONAL R. LOSS FUNCTION ℓ.

otherwise; and if $\mathcal{L} \subseteq \mathbb{R}$, then the mean absolute error is obtained by choosing $\ell(i,j) = |i-j|$ (remark that $\mathcal{L} \subseteq \mathbb{R}$ in itself is no guarantee that the this measure is meaningful).

Flat-line problems. We will only consider so-called **flat-line** problems, i.e. the objects are described by a fixed and finite set of attributes $Q = \{q_i : \Omega \to \mathcal{X}_{q_i} \mid i \in N = \{1,\dots,n\}\}$, and hence any object $a \in \Omega$ can be written as a vector $\mathbf{a} = (q_1(a),\dots,q_n(a))$ in the **data space** $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_{q_i}$. Usually, the search is then concentrated on finding a representation of a classification[1]: $\hat{\lambda} : \mathcal{X} \to \mathcal{L}$. Also, the learning sample Λ is transformed into a set of couples $\Lambda_{\mathcal{X}} = \{\langle \mathbf{a}, d(a)\rangle \mid a \in \mathcal{S}\}$. We will also use the notation $\mathcal{S}_{\mathcal{X}} = \{\mathbf{a} \mid a \in \mathcal{S}\} \subseteq \mathcal{X}$.

FLAT-LINE.

DATA SPACE $\mathcal{X}$.

$\Lambda_{\mathcal{X}}, \mathcal{S}_{\mathcal{X}}$

The stochastic problem. Instead of linking a single class label to an object, the goal is to attach a probability distribution function over the labels to it. In other words, we are looking for a function $\hat{\lambda}_{\text{prob}} : \mathcal{X} \to \mathcal{F}(\mathcal{L})$, where $\mathcal{F}(\mathcal{L})$ is the set of all probability distributions over $\mathcal{L}$. Such classifiers are called **distribution classifiers**. If we write $\mathcal{L} = \{1,\dots,k\}$, an element $f_X \in \mathcal{F}(\mathcal{L})$ is sometimes written as a vector $(\mathcal{P}(X=1),\dots,\mathcal{P}(X=k))$. For $\mathbf{x} \in \mathcal{X}$, we will denote $\hat{\lambda}_{\text{prob}}(\mathbf{x}) = f_{\mathbf{x}}$. Remark that it is possible that the learning sample itself is already stochastic, i.e. $d : \mathcal{S} \to \mathcal{F}(\mathcal{L})$.

$\mathcal{F}(\mathcal{L})$

DISTRIBUTION CLASSIFIER.

$f_{\mathbf{x}}$

Returning a single label. If a single label is asked for an object when a stochastic solution $\hat{\lambda}_{\text{prob}} : \mathcal{X} \to \mathcal{F}(\mathcal{L})$ was found, usually the Bayesian decision (which minimises the risk) is returned, i.e. the label with highest probability.

5.2.2 The ranking problem

The basic problem. For the *complete ranking* (see p. 90) problem, the goal is the same as for classification, but now the labels $\mathcal{L}$ are completely ordered by $\leq_{\mathcal{L}}$ (i.e. $(\mathcal{L}, \leq_{\mathcal{L}})$ is a chain) and $\lambda(a) >_{\mathcal{L}} \lambda(b)$ is to be interpreted as "a is strictly preferred to b". The learning sample now has the form $\Lambda = (\mathcal{S}, (d, \leq_{\mathcal{L}}))$ with $d : \mathcal{S} \to (\mathcal{L}, \leq_{\mathcal{L}})$.

Flat-line problems. For rankings, the objects are not described by a set Q of attributes but by a set C of criteria. A **criterion** [97] (see p. 98) is defined as a mapping $c : \Omega \to (\mathcal{X}_c, \geq_c)$, where $(\mathcal{X}_c, \geq_c)$ is a chain, such that it appears meaningful to compare two objects a and b, according to a particular point of view, on the sole basis of their evaluations $c(a)$ and $c(b)$. We will only consider so-called **true criteria** [20]: a is preferred to b according to criterion c if $c(a) >_c c(b)$. See also Figure 4.5 (see p. 99) for the difference between attributes and criteria.
The search is now concentrated on finding a representation $(\tilde{\lambda}, \leq_{\mathcal{L}})$, where $\tilde{\lambda} : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{L}, \leq_{\mathcal{L}})$. The use of criteria induces an *elementary monotonicity*

CRITERION.

TRUE CRITERION.

[1]To be completely correct, we should write $\hat{\lambda} : \Omega_{\mathcal{X}} \to \mathcal{L}$, with $\Omega_{\mathcal{X}} = \{\mathbf{a} \mid a \in \Omega\} \subseteq \mathcal{X}$.

constraint (see p. 100) on this representation: $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \tilde{\lambda}(\mathbf{x}) \leq_{\mathcal{L}} \tilde{\lambda}(\mathbf{y})$. In other words, the problem boils down to the following:

The flat-line problem

Find a function $\tilde{\lambda} : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{L}, \leq_{\mathcal{L}})$ such that:

(i) $\tilde{\lambda}$ is non-decreasing,

(ii) $\tilde{\lambda}$ minimises some risk functional R.

MONOTONE LEARNING SAMPLE. The learning sample $\Lambda_{\mathcal{X}}$ is called (non-decreasing) **monotone** if $\mathbf{a} \leq_{\mathcal{X}} \mathbf{b} \Rightarrow d(a) \leq_{\mathcal{L}} d(b)$. Remark that in many situations, however, the learning sample $\Lambda_{\mathcal{X}}$ will *not* be monotone itself. There might for example be some error in the sample, or the labels in the learning sample may be based on a different set of criteria than the one used to describe the objects (a more thorough discussion about the reasons of non-monotonicity can be found in Section 4.6.2 (see p. 101), in Section 7.1.3 (see p. 183) the discussion is held in the context of supervised learning).
A popular assumption (typically coming from utility theory[2], e.g. [57, 66] concerning ordinal classification) is that an ordinal variable is the result of a coarsely measured latent continuous variable. In that case, the ordinal ranking problem is altered into a *continuous* ranking problem: find a non-decreasing real function $\tilde{\lambda}_{\mathbb{R}} : \mathcal{X} \to \mathbb{R}$, and, if we write $\mathcal{L} = \{1, \ldots, k\}$, a strictly increasing (extended) real function $U : \{0, 1, \ldots, k\} \to \overline{\mathbb{R}}$ with $U(0) = -\infty$ and $U(k) = +\infty$. The function $\tilde{\lambda} : \mathcal{X} \to \mathcal{L}$ is then defined by $\tilde{\lambda}(\mathbf{x}) = \ell \iff \ell \in]U(\ell - 1), U(\ell)]$. Remind however the discussion in Section 1.2.3. Therefore, we will not pursue this strategy any further. In practice however, as pointed out in [44], a continuous result may be desirable because it allows the division of the population of objects into smaller groups. We will show in Equation (5.2.1) how this can be achieved, if the need arises, for monotone stochastic classifiers.

The stochastic problem. The problem now becomes to find a function $\tilde{\lambda}_{\text{prob}} : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{F}(\mathcal{L}), \trianglelefteq_{(1)})$, where $\trianglelefteq_{(1)}$ is the weak first order stochastic dominance relation. The elementary monotonicity constraint becomes:

$$\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \tilde{\lambda}_{\text{prob}}(\mathbf{x}) \trianglelefteq_{(1)} \tilde{\lambda}_{\text{prob}}(\mathbf{y}) \,.$$

Therefore, the problem can be restated as shown in the frame on the next page. Indeed, let $\mathbf{x} \in \mathcal{X}$. Because of the second and third condition on $\tilde{F}$, the function $\tilde{F}_{\mathbf{x}} : \mathcal{L} \to [0, 1]$ with $\tilde{F}_{\mathbf{x}}(i) := \tilde{F}(\mathbf{x}, i)$, can be interpreted as the cumulative distribution function of a probability distribution $\tilde{f}_{\mathbf{x}}$. If we now define the distribution classifier $\tilde{\lambda}_{\text{prob}} : \mathcal{X} \to \mathcal{F}(\mathcal{L})$ as $\tilde{\lambda}_{\text{prob}}(\mathbf{x}) := \tilde{f}_{\mathbf{x}}$, then the first condition on $\tilde{F}$ makes of $\tilde{\lambda}_{\text{prob}} : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{F}(\mathcal{L}), \trianglelefteq_{(1)})$ a monotone (non-decreasing) function.

[2]See http://cepa.newschool.edu/het/essays/uncert/choiceref.htm for a selection of references.

The stochastic problem

Find a function $\tilde{F} : \mathcal{X} \times \mathcal{L} \to [0,1]$ such that

(i) $\tilde{F}$ is non-increasing in its first argument,

(ii) $\tilde{F}$ is non-decreasing in its second argument,

(iii) $\tilde{F}(\cdot, \max \mathcal{L}) = 1$,

(iv) $\tilde{F}$ minimises some risk functional R.

And define $\tilde{\lambda}_{\text{prob}} : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{F}(\mathcal{L}), \trianglelefteq_{(1)})$ by $\tilde{\lambda}_{\text{prob}}(\mathbf{x}) := \tilde{f}_{\mathbf{x}}$ with cumulative distribution $\tilde{F}(\mathbf{x}, \cdot)$.

In this way, the ordinal ranking problem has been decomposed into $k = |\mathcal{L}|$ *continuous* ranking problems, each with its own derived data set: for each $i \in \mathcal{L}$, find a non-increasing function $\tilde{F}(\cdot, i) : \mathcal{X} \to [0,1]$ based on the learning sample $(\mathcal{S}, (\hat{F}(\cdot, i), \leq))$, where $\hat{F}(\mathbf{x}, i) := \hat{F}_{\mathbf{x}}(i) = \hat{p}\,(\text{class label} \leq_{\mathcal{L}} i \mid \mathbf{x})$, the probability estimated from $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$ that an object a has a class label at most as high as i given that $\mathbf{a} = \mathbf{x}$. However, these k problems can not be treated independent from each other because the corresponding functions $\tilde{F}(\mathbf{x}, \cdot)$ should be non-decreasing!

Returning a single label. Sometimes, it is necessary to return a single label instead of a distribution. However, the Bayesian decision used for distribution classifier comes with a little catch in the present context. Suppose that $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ and $f_{\mathbf{x}} \trianglelefteq_{(1)} f_{\mathbf{y}}$. If we define $\hat{\lambda}'$ as the label with the highest probability, we might end up with $\hat{\lambda}'(\mathbf{x}) >_{\mathcal{L}} \hat{\lambda}'(\mathbf{y})$, i.e. there is no guarantee that $\hat{\lambda}' : (\mathcal{X}, \leq_{\mathcal{X}}) \to (\mathcal{L}, \leq_{\mathcal{L}})$ is still monotone! To illustrate this, consider $\mathcal{L} = \{1, 2, 3, 4\}$, $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ with $f_{\mathbf{x}} = (0.1, 0.3, 0.4, 0.2)$ and $f_{\mathbf{y}} = (0, 0.4, 0.3, 0.3)$. We have indeed that $f_{\mathbf{x}} \trianglelefteq_{(1)} f_{\mathbf{y}}$, but $\hat{\lambda}'(\mathbf{x}) = 3 >_{\mathcal{L}} \hat{\lambda}'(\mathbf{y}) = 2$.
To ensure the monotonicity of the predicted labels, we can perform a step that is *not ordinal in nature*: we take the mean of the distribution assuming equidistance between the labels $\mathcal{L} = \{1, \ldots, k\}$, hence turning the ordinal scale into an interval scale. Next, we round this value to the nearest integer. This then leads to a monotone ranking because it is known [83] that

$$f_X \trianglelefteq_{(1)} f_Y \Rightarrow E[f_X] \leq E[f_Y]\,. \tag{5.2.1}$$

5.2.3 Stochastic representation

Data fitting. The first problem at hand is to find a model that aims at reproducing the given sample data, i.e. a model that is only focussed on attaching the labels to the objects in the learning sample. This is handled by the following theorem, which is a slight reformulation of Theorem 4.7.3 (see p. 108) in the context of supervised learning.

$\hat{f}_{\mathbf{y}}$

$\hat{F}_{\mathbf{y}}$

Theorem 5.2.1. *Let $(d, \leq_{\mathcal{L}})$ be a ranking in $\mathcal{S}$. For all $\mathbf{y} \in \mathcal{S}_{\mathcal{X}} = \{\mathbf{a} \mid a \in \mathcal{S}\}$, let $\hat{f}_{\mathbf{y}}$ denote a probability distribution estimated from $(\mathcal{S}, d)$, i.e. $\hat{f}_{\mathbf{y}}(i) = \hat{p}\,(\text{class} = i \mid \mathbf{y})$. The associated cumulative distribution function is denoted by $\hat{F}_{\mathbf{y}}$. Let $s \in [0,1]$. For all $\mathbf{x} \in \mathcal{X}$, for all $i \in \mathcal{L}$, define:*

$$\boxed{\tilde{F}(\mathbf{x}, i) := (1-s)F_m(\mathbf{x}, i) + sF_M(\mathbf{x}, i)} \tag{5.2.2}$$

where

F_m, F_M

$$F_m(\mathbf{x}, i) = \min_{\mathbf{y} \in (\mathbf{x}] \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i) \qquad \textit{and} \qquad F_M(\mathbf{x}, i) = \max_{\mathbf{y} \in [\mathbf{x}) \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i)\,,$$

$(\mathbf{x}]$, $[\mathbf{x})$

with $(\mathbf{x}] = \{\mathbf{x}' \in \mathcal{X} \mid \mathbf{x}' \leq_{\mathcal{X}} \mathbf{x}\}$ and likewise for $[\mathbf{x})$. If $(\mathbf{x}] \cap \mathcal{S}_{\mathcal{X}} = \emptyset$, define $F_m(\mathbf{x}, i) = 1$, and if $[\mathbf{x}) \cap \mathcal{S}_{\mathcal{X}} = \emptyset$, define $F_M(\mathbf{x}, i) = 0$. It holds that

(i) *$\tilde{F}$ is non-increasing in its first argument,*

(ii) *$\tilde{F}$ is non-decreasing in its second argument and*

(iii) *$\tilde{F}(\cdot, \max \mathcal{L}) = 1$.*

Remark that the previous theorem is in fact a bit more general than Theorem 4.7.3:

- it allows any estimation $\hat{F}_{\mathbf{y}}$ for $F_{\mathbf{y}}$. In this thesis however, we will only use the discrete estimation, i.e.

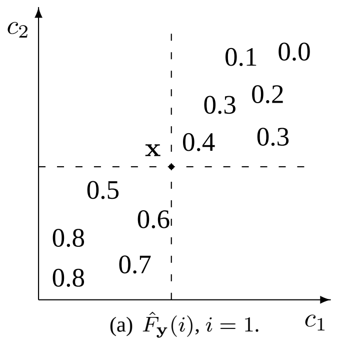
$$\hat{F}_{\mathbf{y}}(i) = \frac{|\{a \in \mathcal{S} \mid (\mathbf{a} = \mathbf{y}) \wedge (d(a) \leq_{\mathcal{L}} i)\}|}{|\{a \in \mathcal{S} \mid (\mathbf{a} = \mathbf{y})\}|}\,.$$

- it defines $\tilde{F}(\mathbf{x}, i)$ for all $\mathbf{x} \in \mathcal{X}$ and not just for $\mathbf{x} \in \mathcal{S}_{\mathcal{X}}$.

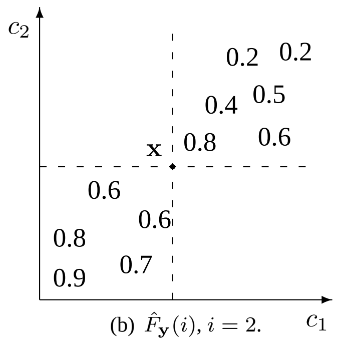
It is easily checked that these adaptations do not affect the proof in any way. Also remark that this theorem allows to work with both ordinal and numerical criteria.

Data extrapolation. After the step of data fitting, a natural continuation is data extrapolation, where we try to predict the label/distribution of unseen vectors in $\mathcal{X} \setminus \mathcal{S}_{\mathcal{X}}$. Since the previous theorem defines probability distributions $\tilde{f}_{\mathbf{x}}$ for all $\mathbf{x} \in \mathcal{X}$, we can simply use the same procedure as for the data fitting, which still leads to a monotone solution (w.r.t. stochastic dominance). This will constitute the essence of the OSDL algorithm.

Example 5.2.2. *Let us have a closer look at what happens exactly. In Figure 5.1, the cumulative distributions $\hat{F}_{\mathbf{y}}(i)$ for $i = 1, 2$ of some learning sample are depicted in two sub-figures (only the data needed to determine $\tilde{F}(\mathbf{x}, i)$ is displayed). Figure 5.1(a) shows the continuous subproblem of finding $\tilde{F}(\cdot, 1)$ where the derived data set is monotone; in (b) the derived data set for the continuous subproblem of finding $\tilde{F}(\cdot, 2)$ is non-monotone. For $i = 1$, we find $F_m(\mathbf{x}, 1) = 0.5$ and $F_M(\mathbf{x}, 1) = 0.4$, and for $i = 2$ we have $F_m(\mathbf{x}, 2) = 0.6$ and $F_M(\mathbf{x}, 2) = 0.8$.*



(a) $\hat{F}_{\mathbf{y}}(i), i = 1.$



(b) $\hat{F}_{\mathbf{y}}(i), i = 2.$

Figure 5.1: Data extrapolation, essence of OSDL.

Interpolation. It should be noted that the above described data extrapolation method based on Equation (5.2.2) has a drawback: it demands that, once fixed, the same parameter s must be used for all vectors. The restrictions this entails are easily demonstrated in the following 1-dimensional example with a monotone sample set: assume we have $\mathcal{S}_{\mathcal{X}} = \{\mathbf{z}_1, \mathbf{z}_2\}$, a single criterion $c : \Omega \to (\{1, \ldots, 4\}, \leq)$ and

(a) Table.

	$\mathbf{z}_1$	$\mathbf{z}_2$	$\mathbf{x}$	$\mathbf{y}$
c	1	4	2	3
$\hat{F}(\cdot, i)$	0.2	0.7		

(b) Linear interpolation.

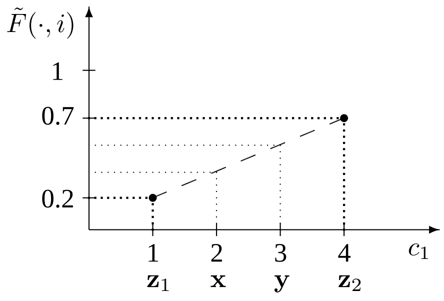


Table 5.1: Simple one-dimensional interpolation.

a continuous ranking $(\hat{F}(\cdot, i), \leq)$ with $\hat{F}(\cdot, i) : \mathcal{S}_{\mathcal{X}} \to [0, 1]$, as shown in Table 5.1. We find $F_m(\mathbf{x}, i) = F_m(\mathbf{y}, i) = \hat{F}(\mathbf{z}_1, i) = 0.2$ and $F_M(\mathbf{x}, i) = F_M(\mathbf{y}, i) = \hat{F}(\mathbf{z}_2, i) = 0.7$, and therefore, by Equation (5.2.2), $\tilde{F}(\mathbf{x}, i) = \tilde{F}(\mathbf{y}, i)$, even if we know that $\mathbf{x} <_{\mathcal{X}} \mathbf{y}$. Simple linear interpolation (if we forget about the ordinal nature of the problem) makes a difference between $\mathbf{x}$ and $\mathbf{y}$.

An idea that comes to mind, is to apply Theorem 5.2.1 on the given sample (i.e. data fitting ensuring monotonicity), and afterwards, use some standard interpolation technique on the cumulative distribution functions, such as triangle-based interpolation or natural neighbour interpolation [120]. But this doesn't work: first of all, this would force us to transform the ordinal scales on the axes to an interval scale. More importantly however, even if such a transformation could be done, the existing interpolation techniques cannot guarantee a monotone behaviour, making them ill suited for the job. This can be seen from the following example:

Example 5.2.3. *Assume we have* $\mathcal{S}_{\mathcal{X}} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$, $\mathcal{X} = \{0,1,2\} \times \{0,1,2\}$ *and a continuous ranking* $(F(\cdot, i), \leq)$ *with* $F(\cdot, i) : \mathcal{S}_{\mathcal{X}} \to [0,1]$, *as shown in Table 5.2. We want an interpolation for* $\mathbf{x} = (1,1)$. *Because* $\tilde{F}$ *must be non-increasing in its*

(a) Table.

	$\mathbf{z}_1$	$\mathbf{z}_2$	$\mathbf{z}_3$	$\mathbf{x}$
$\mathcal{X}$	(0,1)	(2,0)	(1,2)	(1,1)
$F(\cdot, i)$	1	0	1	

(b) Interpolation.

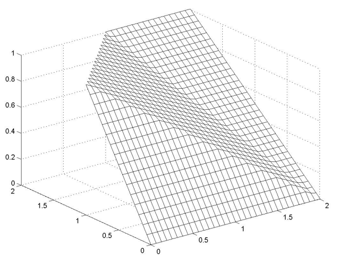


Table 5.2: Example of non-monotone interpolation.

first argument, and $\mathbf{z}_1 \leq_{\mathcal{X}} \mathbf{x} \leq_{\mathcal{X}} \mathbf{z}_3$, *we know that* $\tilde{F}(x,i) = 1$. *But interpolation would result in* $\tilde{F}(x,i) < 1$, *since it will be evaluated as* $w_1 \times 1 + w_2 \times 0 + w_3 \times 1$, *with* $w_2 \neq 0$.

Very recently, a monotone interpolation method based on splines was proposed in [8]. It would be worthwhile to investigate this path further since it is most likely that it will boost the performance of the data extrapolation.

5.3 The algorithm

5.3.1 OSDL

Introduction. Based on Theorem 5.2.1, we can easily create an instance-based learner. The learner is built in 2 phases: first the data base is constructed, simply keeping track of the discrete (cumulative) distribution estimates, then the parameter s is determined via leave-one-out cross validation (similar to how the parameter k is found in the WEKA [124] implementation of the k-Nearest Neighbour method). Classifying a new instance $\mathbf{x} \in \mathcal{X}$ is done by applying Equation (5.2.2).

The basic algorithm. If we want the distribution for a new instance, we use Algorithm 5.1. If a single label is required, the function **singleLabel** returns the mean[3] – rounded to the nearest integer – of a probability distribution (see (5.2.1)).

[3]Remember that this is in fact not a valid ordinal operation. However, using the "ordinal-proof" median (the quantile of order 1/2) led to extremely poor results on the first tests, so we abandoned it quite rapidly.

Algorithm 5.1 : distributionFor, calculate the probability distribution

```
$\mathbf{x} \leftarrow$ new instance to classify;
calculate the bounds $F_m(\mathbf{x}, \cdot)$ and $F_M(\mathbf{x}, \cdot)$;
return $(1 - s)F_m(\mathbf{x}, \cdot) + sF_M(\mathbf{x}, \cdot)$;
```

The algorithm for building the data base is shown in Algorithm 5.2. The data base can be updated when new data becomes available. For an example $\langle \mathbf{x}, i \rangle \in \Lambda_{\mathcal{X}}$, the function **addInstance**($\langle \mathbf{x}, i \rangle$) simply adds $\mathbf{x}$ to the data base if it is not already present, and updates the relative frequency $\hat{f}_{\mathbf{x}} \in \mathcal{F}(\mathcal{L})$ of examples with vector representation $\mathbf{x}$. The function **removeInstance** just does the opposite. As always, we denote probability distributions by f and the corresponding cumulative distribution functions by F. To fill in the role of the risk functional R that has to be minimised, we have chosen the mean of the losses over all $a \in \mathcal{S}$: $R = E[\ell(d(a), \text{prediction}(a))]$. Algorithm 5.2 builds the data base and performs a leave-one-out cross validation to tune the parameter s.

Scalability. It is fairly simple to perform all necessary calculations in a parallel way by simply dividing the training sample over n machines: all calculations are based on taking the minimum and the maximum in $\mathbb{R}$, which are decomposable aggregation operators (for two sets $A, B \subseteq \mathbb{R}$, we have $\min A \cup B = \min(\min A, \min B)$ and likewise for the maximum).

A possibly more efficient approach. Each time a new instance has to be ranked, the previous algorithm asks to calculate the values $F_m(\mathbf{x}, i) = \min_{\mathbf{y} \in (\mathbf{x}] \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i)$ and $F_M(\mathbf{x}, i) = \max_{\mathbf{y} \in [\mathbf{x}) \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i)$ for all $i \in \mathcal{L}$. This is clearly the bottleneck of this algorithm because time and again, it demands going over the whole data base. Probably[4], a better organisation of the data base would reduce the computation time. A possible technique is the following: cluster the vectors from $\mathcal{S}_{\mathcal{X}}$ into a series of intervals $[\mathbf{y}_\ell, \mathbf{y}_r]$. If a new instance $\mathbf{x}$ must be ranked, we know that for some interval $[\mathbf{y}_\ell, \mathbf{y}_r]$:

(i) If $\mathbf{x}$ is incomparable with both $\mathbf{y}_\ell$ and $\mathbf{y}_r$, then $\mathbf{x}$ is also incomparable to all vectors $\mathbf{y} \in [\mathbf{y}_\ell, \mathbf{y}_r]$. This means we can disregard the vectors in $[\mathbf{y}_\ell, \mathbf{y}_r]$ for the calculation of $F_m(\mathbf{x}, \cdot)$ and $F_M(\mathbf{x}, \cdot)$.

(ii) If $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}_\ell$ then $(\forall \mathbf{y} \in [\mathbf{y}_\ell, \mathbf{y}_r])(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y})$, which means that by keeping in memory $F'_{\mathbf{y}_\ell} = \max_{\mathbf{y} \in [\mathbf{y}_\ell, \mathbf{y}_r]} \hat{F}_y$ allows us to disregard the vectors in $[\mathbf{y}_\ell, \mathbf{y}_r]$ and only consider $F'_{\mathbf{y}_\ell}$ for the calculation of $F_M(\mathbf{x}, \cdot)$.

(iii) If $\mathbf{x} \geq_{\mathcal{X}} \mathbf{y}_r$ then $(\forall \mathbf{y} \in [\mathbf{y}_\ell, \mathbf{y}_r])(\mathbf{x} \geq_{\mathcal{X}} \mathbf{y})$, which makes it interesting to memorise $F'_{\mathbf{y}_r} = \min_{\mathbf{y} \in [\mathbf{y}_\ell, \mathbf{y}_r]} \hat{F}_y$.

[4]We have not yet tested this idea by practical experience. This is a path for future research.

Algorithm 5.2 : Build, build the data base and tune the parameter s with leave-one-out cross validation

```
// commit data base to memory
for all examples $\langle \mathbf{x}, i \rangle \in \Lambda_{\mathcal{X}}$ do
  addInstance($\langle \mathbf{x}, i \rangle$);
end for
if $\min \mathcal{X} \notin$ data base then
  // to make sure $F_m$ can always be calculated
  addInstance($\langle \min \mathcal{X}, \min \mathcal{L} \rangle$);
end if
if $\max \mathcal{X} \notin$ data base then
  // to make sure $F_M$ can always be calculated
  addInstance($\langle \max \mathcal{X}, \max \mathcal{L} \rangle$);
end if

// cross validation for parameter tuning
for all examples $\langle \mathbf{x}, i \rangle \in \Lambda_{\mathcal{X}}$ do
  removeInstance($\langle \mathbf{x}, i \rangle$);
  calculate the bounds $F_m(\mathbf{x}, \cdot)$ and $F_M(\mathbf{x}, \cdot)$;
  // we check $s = 0$ to $1$ with steps of $0.1$
  for $s' = 0$ to $s' = 10$ do
    $s \leftarrow s'/10$;
    $F_s(\mathbf{x}, \cdot) \leftarrow (1-s)F_m(\mathbf{x}, \cdot) + sF_M(\mathbf{x}, \cdot)$;
    error$[s']$ = error$[s']$+ loss ($i$, singleLabel($f_s(\mathbf{x}, \cdot)$));
  end for
  addInstance($\langle \mathbf{x}, i \rangle$);
end for
for $s' = 0$ to $s' \leq 10$ do
  $R[s'] \leftarrow$ error$[s']/\,|\Lambda_{\mathcal{X}}|$;
end for
$s \leftarrow (\arg\min_{s'=0,\dots,10} R[s'])/10$;
```

To find these clusters, a grid-based clustering technique [14], like MAFIA [81], is probably most suited since this technique is naturally linked to intervals in $\mathcal{X}$. Investing some more energy in the building of the data base will then save time in the classification process.

5.3.2 Balanced OSDL

Introduction. We already mentioned the rigidness of fixing globally a single parameter s. It woud be interesting to have a more locally adaptive parameter that incorporates more of the available information. When dealing with non-monotone data sets, this inflexible behaviour is even more annoying. Indeed, consider the following extreme case: we have a data set $\Lambda_{\mathcal{X}}$ with $\min \mathcal{X} \notin \mathcal{S}_{\mathcal{X}}$ and $\max \mathcal{X} \notin \mathcal{S}_{\mathcal{X}}$. Now add two serious outliers to this data set: $\langle \min \mathcal{X}, \max \mathcal{L} \rangle$ and $\langle \max \mathcal{X}, \min \mathcal{L} \rangle$. In that case we find for all $\mathbf{x} \in \mathcal{X}$ and for all $i <_{\mathcal{L}} \max \mathcal{L}$ that $F_m(\mathbf{x}, i) = 0$ and $F_M(\mathbf{x}, i) = 1$ (see Table 5.3). Therefore, we always have that $\tilde{F}(\mathbf{x}, i) = s$.

	a_1	a_2	a_3	a_4	a_5	a_6
c	1	2	3	4	5	6
d	2	1	1	2	2	1
$\hat{F}(\cdot, 1)$	0	1	1	0	0	1
$\tilde{F}_{\text{OSDL}}(\cdot, 1)$	0.5	0.5	0.5	0.5	0.5	0.5

Table 5.3: An extreme situation. OSDL with $s = \frac{1}{2}$.

In this section, we try to overcome this latter problem by extracting additional information from the data set: we construct a weighting mechanism that considers the number of samples that endorse the idea that a vector $\mathbf{x}$ belongs to some rank $\leq_{\mathcal{L}} i$, and the number of samples that contradict this. The proposed adaptation only deals with non-monotone behaviour, in case the data set is monotone, we still end up with the standard OSDL technique.

Recapitulation Section 4.7.4

Weighting. Assume the ranking $(\lambda, \leq_{\mathcal{L}})$ is known and that we are in the ideal situation where the probability distributions $f_{\mathbf{x}}$ behave monotonically, i.e. we could not have chosen a better set of criteria to describe the problem. This means that for all $\mathbf{y} \leq_{\mathcal{X}} \mathbf{x}$, it holds that $f_{\mathbf{y}} \trianglelefteq_{(1)} f_{\mathbf{x}}$ or $F_{\mathbf{y}}(i) \geq F_{\mathbf{x}}(i)$ for all $i \in \mathcal{L}$, and therefore

$$F_{\mathbf{x}}(i) \leq \min_{\mathbf{y} \in (\mathbf{x}]} F_{\mathbf{y}}(i) .$$

At the same time, for all $\mathbf{y} \geq_{\mathcal{X}} \mathbf{x}$, it should hold that $f_{\mathbf{x}} \trianglelefteq_{(1)} f_{\mathbf{y}}$, or

$$F_{\mathbf{x}}(i) \geq \max_{\mathbf{y} \in [\mathbf{x})} F_{\mathbf{y}}(i) .$$

Now consider the case where we only have a learning sample $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$ at our disposal. If it is a good sample, it should reflect the actual ranking $(\lambda, \leq_{\mathcal{L}})$, so the estimations $\hat{f}_{\mathbf{x}}$ of the probability distributions $f_{\mathbf{x}}$ should also be monotone. In that case, we should have

$$\min_{\mathbf{y} \in (\mathbf{x}] \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i) \;=\; F_m(\mathbf{x}, i) \;\geq\; \tilde{F}_{\mathbf{x}}(i) \;\geq\; F_M(\mathbf{x}, i) \;=\; \max_{\mathbf{y} \in [\mathbf{x}) \cap \mathcal{S}_{\mathcal{X}}} \hat{F}_{\mathbf{y}}(i) .$$

So we can interpret $F_m(\mathbf{x}, i)$ as the one pushing $\mathbf{x}$ towards labels $>_{\mathcal{L}} i$, and similarly $F_M(\mathbf{x}, i)$ as the one pulling $\mathbf{x}$ down to labels $\leq_{\mathcal{L}} i$. Indeed, as $F_m(\mathbf{x}, i)$ drops to 0, $\tilde{F}_{\mathbf{x}}(i)$ is also forced towards 0, implying that the probability that $\mathbf{x}$ gets a label at most i plunges to zero: $(\forall \ell \leq_{\mathcal{L}} i)(\tilde{f}_{\mathbf{x}}(\ell) \to 0)$. In other words, the label attached to $\mathbf{x}$ is very likely higher than i. Similarly, as $F_M(\mathbf{x}, i)$ rockets towards 1, it also

lifts $\tilde{F}_{\mathbf{x}}(i)$ up to the same level. This implies that the probability that $\mathbf{x}$ gets a label at most i jumps to one: $(\forall \ell \leq_{\mathcal{L}} i)(\tilde{f}_{\mathbf{x}}(\ell) \to 1)$.
Sometimes, however, $F_m(\mathbf{x}, i)$ and $F_M(\mathbf{x}, i)$ may become a bit overactive and pull and push $\mathbf{x}$ over the edge. Indeed, it is possible that $F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i)$. When this happens, we could look for evidence supporting the pushing, and for evidence supporting the pulling. In more concrete terms: we can regard the objects $a \in \mathcal{S}$ with $\mathbf{a} \leq_{\mathcal{X}} \mathbf{x}$ and labelled $\lambda(a) >_{\mathcal{L}} i$ as evidence in favour of the pushy $F_m(\mathbf{x}, i)$, and objects b with $\mathbf{b} \geq_{\mathcal{X}} \mathbf{x}$ labelled $\lambda(b) \leq_{\mathcal{L}} i$ as being in league with the pulling $F_M(\mathbf{x}, i)$.

Proposition 5.3.1. *Let $\mathbf{x} \in \mathcal{X}$ and denote for all $i \in \mathcal{L}$*

N_m, N_M

$$\begin{aligned} N_m(\mathbf{x}, i) &= \left|\{a \in \mathcal{S} \mid (\mathbf{a} \leq_{\mathcal{X}} \mathbf{x}) \wedge (d(a) >_{\mathcal{L}} i)\}\right|, \\ N_M(\mathbf{x}, i) &= \left|\{a \in \mathcal{S} \mid (\mathbf{a} \geq_{\mathcal{X}} \mathbf{x}) \wedge (d(a) \leq_{\mathcal{L}} i)\}\right|. \end{aligned}$$

Now define $\tilde{F}(\mathbf{x}, i)$ as in Equation (5.2.2) if $F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)$, and otherwise as

$$\boxed{\tilde{F}(\mathbf{x}, i) = \frac{N_m(\mathbf{x}, i)F_m(\mathbf{x}, i) + N_M(\mathbf{x}, i)F_M(\mathbf{x}, i)}{N_m(\mathbf{x}, i) + N_M(\mathbf{x}, i)}} \tag{5.3.1}$$

It holds that

(i) *$\tilde{F}$ is non-increasing in its first argument,*

(ii) *$\tilde{F}$ is non-decreasing in its second argument and*

(iii) $\tilde{F}(\cdot, \max \mathcal{L}) = 1$.

Proof.
First remark that if $F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i)$, then $N_m(\mathbf{x}, i) > 1$ and $N_M(\mathbf{x}, i) > 1$. Indeed, assume $N_m(\mathbf{x}, i) = 0$, this means that all training examples a with $\mathbf{a} \leq_{\mathcal{X}} \mathbf{x}$ have a label $\leq_{\mathcal{L}} i$. This implies that for these objects $F_{\mathbf{a}}(i) = 1$, whence $F_m(\mathbf{x}, i) = 1$, a contradiction. $N_M(\mathbf{x}, i) = 0$ leads in a similar fashion to a contradiction.
Now consider $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$, implying $(\mathbf{x}] \subseteq (\mathbf{y}]$ and $[\mathbf{x}) \supseteq [\mathbf{y})$ (see Figure 4.6, p. 105). We therefore have that $F_m(\mathbf{x}, i) \geq F_m(\mathbf{y}, i)$ and $F_M(\mathbf{x}, i) \geq F_M(\mathbf{y}, i)$.

(i) We first show that $\tilde{F}$ is non-increasing in its first argument.

(a) $F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i)$ and $F_m(\mathbf{y}, i) \geq F_M(\mathbf{y}, i)$. We must show that $\tilde{F}_{\mathbf{x}}(i)$ defined via Equation (5.3.1) is at least as big as $\tilde{F}_{\mathbf{y}}(i)$ defined via Equation (5.2.2), for all $s \in [0, 1]$. This follows directly from $F_M(\mathbf{y}, i) \leq F_m(\mathbf{y}, i) \leq F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i)$.

(b) $F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)$ and $F_m(\mathbf{y}, i) < F_M(\mathbf{y}, i)$. In this case, we have $F_m(\mathbf{y}, i) < F_M(\mathbf{y}, i) \leq F_M(\mathbf{x}, i) \leq F_m(\mathbf{x}, i)$.

(c) $F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)$ and $F_m(\mathbf{y}, i) \geq F_M(\mathbf{y}, i)$. To prove the present case, we rely on the following property: Let $a_1, a_2, b_1, b_2 \in \mathbb{R}$, with $a_1 \geq a_2$ and $b_1 \geq b_2$. Let $s_1, s_2 \in [0, 1]$. If $s_1 \leq s_2$, then (see also Figure 5.2)

$$A = s_1\, a_1 + (1 - s_1)\, b_1 \;\geq\; B = s_2\, a_2 + (1 - s_2)\, b_2\,.$$

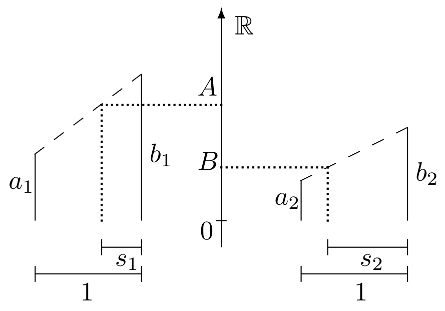


Figure 5.2: Interpolation for Balanced OSDL

Therefore, if we can show that

$$\frac{N_m(\mathbf{x}, i)}{N_m(\mathbf{x}, i) + N_M(\mathbf{x}, i)} \leq \frac{N_m(\mathbf{y}, i)}{N_m(\mathbf{y}, i) + N_M(\mathbf{y}, i)}\,,$$

we will have completed the proof .

It is clear that $N_m(\mathbf{x}, i) \leq N_m(\mathbf{y}, i)$ (because $(\mathbf{x}] \subseteq (\mathbf{y}]$) and $N_M(\mathbf{y}, i) \leq N_M(\mathbf{x}, i)$. Hence

$$N_m(\mathbf{x}, i) N_M(\mathbf{y}, i) \leq N_m(\mathbf{y}, i) N_M(\mathbf{x}, i)\,.$$

Adding to both sides the term $N_m(\mathbf{x}, i) N_m(\mathbf{y}, i)$ leads to

$$N_m(\mathbf{x}, i)(N_m(\mathbf{y}, i) + N_M(\mathbf{y}, i)) \leq N_m(\mathbf{y}, i)(N_m(\mathbf{x}, i) + N_M(\mathbf{x}, i))\,,$$

completing the proof.

(ii) We now proceed by showing that $\tilde{F}$ is non-decreasing in its second argument. The proof is very similar to the previous one. If $i <_{\mathcal{L}} j$, then $F_m(\mathbf{x}, i) \leq F_m(\mathbf{x}, j)$ and $F_M(\mathbf{x}, i) \leq F_M(\mathbf{x}, j)$.

(a) $F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i)$ and $F_m(\mathbf{x}, j) \geq F_M(\mathbf{x}, j)$. In this case, we have that $F_m(\mathbf{x}, i) < F_M(\mathbf{x}, i) \leq F_M(\mathbf{x}, j) \leq F_m(\mathbf{x}, j)$.

(b) $F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)$ and $F_m(\mathbf{x}, j) < F_M(\mathbf{x}, j)$. In this case, we have that $F_M(\mathbf{x}, i) \leq F_m(\mathbf{x}, i) \leq F_m(\mathbf{x}, j) < F_M(\mathbf{x}, j)$.

(c) $F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)$ and $F_m(\mathbf{y}, i) \geq F_M(\mathbf{y}, i)$. We have that $N_m(\mathbf{x}, j) \leq N_m(\mathbf{x}, i)$ (because $d(a) > j \Rightarrow d(a) > i$) and $N_M(\mathbf{x}, i) \leq N_M(\mathbf{x}, j)$. This leads to

$$\frac{N_m(\mathbf{x}, j)}{N_m(\mathbf{x}, j) + N_M(\mathbf{x}, j)} \leq \frac{N_m(\mathbf{x}, i)}{N_m(\mathbf{x}, i) + N_M(\mathbf{x}, i)}\,,$$

completing the proof. □

The following lemma shows that we can incorporate a second parameter besides s into the Balanced OSDL variant:

Lemma 5.3.2. *Let $s' \in [0,1]$. The previous proposition still holds if we replace Equation (5.3.1) by*

$$\tilde{F}(\mathbf{x}, i) = \frac{(1-s')\,N_m(\mathbf{x}, i)F_m(\mathbf{x}, i) + s'\,N_M(\mathbf{x}, i)F_M(\mathbf{x}, i)}{(1-s')N_m(\mathbf{x}, i) + s'N_M(\mathbf{x}, i)}\,.$$

Proof.
For $s' = 0$ or $s' = 1$, this statement is obvious. Assume $s' \in]0,1[$. It holds that $N_m(\mathbf{x}, i)N_M(\mathbf{y}, i) \leq N_m(\mathbf{y}, i)N_M(\mathbf{x}, i)$. Multiplying with the positive value $s'(1-s')$ and adding the term $(1-s')^2 N_m(\mathbf{x}, i)N_m(\mathbf{y}, i)$ to both sides leads to

$$\begin{aligned}(1-s')N_m(\mathbf{x}, i)\,[(1-s')N_m(\mathbf{y}, i) + s'N_M(\mathbf{y}, i)] \leq \\ (1-s')N_m(\mathbf{y}, i)\,[(1-s')N_m(\mathbf{x}, i) + s'N_M(\mathbf{x}, i)]\,,\end{aligned}$$

which proves the lemma. □

If $s' = 0.5$, we recover Proposition 5.3.1.

Example 5.3.3. *Consider again Table 5.3. We can now calculate the balanced variant. If we look at $\mathbf{a}_3$ for example, we still find that $F_m(\mathbf{a}_3, 1) = 0$ and $F_M(\mathbf{a}_3, 1) = 1$, but also that $N_m(\mathbf{a}_3, 1) = |\{a_1\}| = 1$ and $N_M(\mathbf{a}_3, 1) = |\{a_3, a_6\}| = 2$. Therefore, according to Equation (5.3.1), we find that $\tilde{F}_{\text{B-OSDL}}(\cdot, 1) = (1 \times 0 + 2 \times 1)/(1+2) = 0.66$.*

		a_1	a_2	a_3	a_4	a_5	a_6
c		1	2	3	4	5	6
d		2	1	1	2	2	1
$\tilde{F}_{\text{OSDL}}(\cdot, 1)$	$(s = 0.5)$	0.5	0.5	0.5	0.5	0.5	0.5
$\tilde{F}_{\text{B-OSDL}}(\cdot, 1)$	$(s' = 0.75)$	0.9	0.9	0.857	0.6	0.5	0.5
$\tilde{F}_{\text{B-OSDL}}(\cdot, 1)$	$(s' = 0.5)$	0.75	0.75	0.66	0.33	0.25	0.25
$\tilde{F}_{\text{B-OSDL}}(\cdot, 1)$	$(s' = 0.25)$	0.5	0.5	0.4	0.143	0.1	0.1

Table 5.4: OSDL versus Balanced OSDL (B-OSDL).

Scalability. Besides the minimum and maximum operators, the Balanced OSDL algorithm also needs some additional counts. Again, these can easily be decomposed and spread out over several processors.

5.3.3 Summary

<u>OSDL</u>:
for $s \in [0,1]$, define the cumulative distribution of $\mathbf{x} \in \mathcal{X}$ as

$$\tilde{F}(\mathbf{x}, i) := (1-s)F_m(\mathbf{x}, i) + sF_M(\mathbf{x}, i)\,.$$

<u>Balanced OSDL</u>:
for $s, s' \in [0,1]$, define the cumulative distribution of $\mathbf{x} \in \mathcal{X}$ as

$$\tilde{F}(\mathbf{x}, i) := \begin{cases} (1-s)F_m(\mathbf{x}, i) + sF_M(\mathbf{x}, i) & \text{, if } F_m(\mathbf{x}, i) \geq F_M(\mathbf{x}, i)\,, \\ \frac{(1-s')\,N_m(\mathbf{x},i)F_m(\mathbf{x},i) + s'\,N_M(\mathbf{x},i)F_M(\mathbf{x},i)}{(1-s')N_m(\mathbf{x},i) + s'N_M(\mathbf{x},i)} & \text{, otherwise.} \end{cases}$$

Where

$$F_m(\mathbf{x}, i) = \min_{\mathbf{y} \in (\mathbf{x}] \cap \mathcal{S}_\mathcal{X}} \hat{F}_\mathbf{y}(i)\,, \qquad F_M(\mathbf{x}, i) = \max_{\mathbf{y} \in [\mathbf{x}) \cap \mathcal{S}_\mathcal{X}} \hat{F}_\mathbf{y}(i)\,,$$

$$\hat{F}_\mathbf{y}(i) = \frac{|\{a \in \mathcal{S} \mid (\mathbf{a} = \mathbf{y}) \wedge (d(a) \leq_\mathcal{L} i)\}|}{|\{a \in \mathcal{S} \mid (\mathbf{a} = \mathbf{y})\}|}\,,$$

$$N_m(\mathbf{x}, i) = |\{a \in \mathcal{S} \mid (\mathbf{a} \leq_\mathcal{X} \mathbf{x}) \wedge (d(a) >_\mathcal{L} i)\}|\,,$$

$$N_M(\mathbf{x}, i) = |\{a \in \mathcal{S} \mid (\mathbf{a} \geq_\mathcal{X} \mathbf{x}) \wedge (d(a) \leq_\mathcal{L} i)\}|\,.$$

A monotone deterministic labelling $\tilde{\lambda} : \mathcal{X} \to \mathcal{L}$ can be obtained by defining

$$\tilde{\lambda}(\mathbf{x}) := E[\tilde{f}_\mathbf{x}]\,.$$

This last calculation is, however, non-ordinal.

5.4 Experiments

5.4.1 Generating artificial data

Introduction. We need two kinds of learning/test samples $\Lambda = (\mathcal{S}, (d, \leq_{\mathcal{L}}))$ for our experiments: monotone data sets and non-monotone data sets that nevertheless reflect a monotone idea. Once we have a monotone data set at our disposal, we will show there are several ways of transforming it into a non-monotone one simulating different problems that might occur in real data sets.

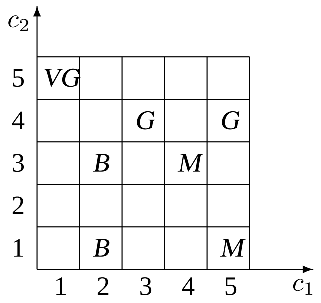
Monotone (non-stochastic) data sets. In [89], Potharst described an algorithm for generating monotone test samples for finite data spaces. There are only two minor drawbacks: not all monotone samples are generated with the same probability and it does not allow to generate full monotone functions when the data space becomes to big (because the method needs to store all vectors into memory). Nevertheless, it is the only one, and it is far from easy to overcome both mentioned drawbacks.

Non-monotone data sets. In Section 4.6.2 (see p. 101) we discussed possible scenarios for the occurrence of *reversed preference* (see p. 100) (objects not complying with the monotonicity demand). The first reason was that the set C of criteria describing the objects is not complete, i.e. that some essential criteria are missing to obtain a monotone representation. The second reason was that there are some errors in the labelling. The third reason was that the labels for the learning examples came from different sources.
Assume we have a monotone learning sample $\Lambda_{\mathcal{X}}$. The second and third reason for reversed preference can both be simulated by adding some noise to the function d in the learning sample (not in the test sample however). The absence of certain criteria can be simulated by simply omitting some criteria from the set C, i.e. by projecting $\mathcal{X}$ on a lower dimensional subspace $\mathcal{X}' \subset \mathcal{X}$ and continue with the learning sample $\Lambda_{\mathcal{X}'}$. Remark that the data sets resulting from such a projection can lead to a quite difficult learning task: even omitting one single axis may cause extreme reversed preferences as can be seen in Figure 5.3. In the experiments, we will only consider reversed preferences introduced by projection, because this is a harder learning task than simply adding some controlled error.

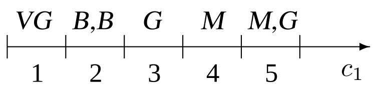
5.4.2 Methods used in the experiments

Instance-based Learners.

NON-MONOTONE. Since the proposed algorithm is an instance-based learner, we include a k-nearest neighbour (k-**NN**) algorithm in our comparisons. We have taken the freely available WEKA implementation [124]: `weka.classifiers.lazy.IBk`, where we used the option of tuning the parameter k by leave-one-out optimisation.



(a) Monotone data set.



(b) Projected data set.

Figure 5.3: Leaving out one axis from the monotone data set. $\mathcal{L} = \{$*B(ad)*, *M(oderate)*, *G(ood)*, *V(ery) G(ood)*$\}$

MONOTONE. We also include Ben-David's instance-based learner **OLM**, Ordinal Learning Model [9], which was the first algorithm specifically adapted to deal with rankings. It should be commented that in spite of its name, some techniques used in this algorithm are non-ordinal. Since the result of OLM can be numeric, we rounded the result to the nearest label, where, as in OLM, equidistance was assumed to transform the ordinal scale into an interval scale. Lastly, OLM can in fact deliver non-monotone results, but this can be easily resolved (Section 3.1.1 (see p. 53)). The implementation used in the comparisons is however the algorithm as described in [9].

Decision Trees.

NON-MONOTONE. One of the usual methods used for comparison is C4.5 [93]. Again, we relied on the WEKA implementation: `weka.classifiers.trees.j48.J48`. We altered it a bit to be able to deal with ordinal attributes, we declared the attributes nominal but took only splits of the form $c(\cdot) \leq_c v$ into account. We consider both pruned and unpruned trees.

MONOTONE. Classification trees are however not adapted for ranking problems. Potharst [89] describes the algorithm **MDT**, Monotone Decision Trees, for generating monotone decision trees based on monotone (non-stochastic) data. MDT is an adaptation of the algorithm described in [74] which only handles binary rankings.

Minimal ($\lambda_{\min}$) and maximal ($\lambda_{\max}$) extensions.

MONOTONE. Denote $\mathcal{L} = \{1, \ldots, k\}$. The most simple manner to rank an instance $\mathbf{x}$ is to assign it the minimal label $\lambda_{\min}(\mathbf{x})$ such that $(\forall \mathbf{y} \leq_{\mathcal{X}} \mathbf{x})(\lambda_{\min}(\mathbf{x}) \geq_{\mathcal{L}} \lambda(\mathbf{y}))$, i.e. $\lambda_{\min}(\mathbf{x}) = \max\{d(a) \mid a \in \mathcal{S} \wedge \mathbf{a} \leq_{\mathcal{X}} \mathbf{x}\}$, or 1 if the set $\{a \in \mathcal{S} \wedge \mathbf{a} \leq_{\mathcal{X}} \mathbf{x}\}$ is empty. Another option is to assign it the maximal label $\lambda_{\max}(\mathbf{x})$ such that $(\forall \mathbf{y} \geq_{\mathcal{X}} \mathbf{x})(\lambda_{\max}(\mathbf{x}) \leq_{\mathcal{L}} \lambda(\mathbf{y}))$, i.e. $\lambda_{\max}(\mathbf{x}) = \min\{d(a) \mid a \in \mathcal{S} \wedge \mathbf{a} \geq_{\mathcal{X}} \mathbf{x}\}$, or k if the set $\{a \in \mathcal{S} \wedge \mathbf{a} \geq_{\mathcal{X}} \mathbf{x}\}$ is empty.

$\lambda_{\min}(\mathbf{x})$

$\lambda_{\max}(\mathbf{x})$

It is clear that both methods lead to a monotone classifier. In the tables, we denote these algorithms by **Min** and **Max**.

Statistical.

NON-MONOTONE. To include a totally different learner, we also add the **Naive Bayes** method, using the WEKA implementation `weka.classifiers.bayes.NaiveBayes`.

OSDL and Balanced OSDL.

MONOTONE. Remark that, for monotone non-stochastic data, OSDL with parameter $s = 0$ coincides with the minimal extension, and that $s = 1$ leads to the maximal extension. Hence, with the tuning of s, it is very likely that OSDL will be at least as good as any of these two methods. In the binary case, $|\mathcal{L}| = 2$, OSDL will always be equal to one of these two extensions (remind that this only holds for non-stochastic data), but as $|\mathcal{L}|$ grows, intermediate extensions become possible.
The Balanced OSDL algorithm used in the experiments does not yet incorporate the second parameter s' (i.e. the constant $s' = 0.5$ is used).

5.4.3 Performance measures

A note on measures. There does not really exist a measure that is especially developed for ordinal ranking problems. The perfect measure would be a mix of (1) how close the predicted label is to the actual label (which is rather difficult to measure for ordinal classes), (2) accuracy and (3) monotonicity.

STANDARD MEASURES FOR PERFORMANCE. The *mean square error* MSE and *mean absolute error* MAE both measure how close the predictions are to the actual class labels, but they are not really applicable to ordinal problems because they assume an interval scale. Still, they do give a clue of the performance. The classical *accuracy* (i.e. percentage of correctly classified instances) is valid on ordinal problems, but does not punish severe faults such as ranking a class-1 instance as belonging to class 5. A known measure for ordinal problems is *Kendall's tau* (see p. 53), defined as

$$\tau = \frac{P - Q}{P + Q} \in [-1, 1] \,,$$

where P is the number of concordant pairs[5] $(a, b) \in \mathcal{S}_{\text{test}} \times \mathcal{S}_{\text{test}}$, and Q the number of discordant pairs (rank reversal). It is however harder to interpret correctly than the others.

MONOTONICITY. None of these measures incorporate the idea of monotonicity. Even more, they tend to punish monotone classifiers: consider for example Figure 5.3(b), then a monotone classifier is forced to assign at least 3 objects to the wrong label, non-monotone classifiers on the other hand are only forced to assign 1 object to the wrong label. So the monotonicity constraint has a negative impact on the previous measures.

[5] A pair (a, b) is called concordant if the order between $d(a)$ and $d(b)$ is maintained between predicted(a) and predicted(b).

It is possible to define a *degree of monotonicity* as

$$\frac{\#\text{ monotone couples in } \mathcal{S}_{\text{test}}}{\#\text{ couples in } \mathcal{S}_{\text{test}}}\,.$$

However, monotonicity is a property of a classifier, so either the classifier is monotone, or it is not[6]. Either the user demands monotonicity, and rejects any classifier that may produce non-monotone results, or the user is not interested whether or not the classifier is monotone, as long as it delivers good performance. If only performance is strived for, modelers building the classifier may come to the conclusion that a higher monotonicity degree leads to better results (as we may expect in our experiments with monotone non-stochastic data), and may therefore aim at optimising this measure. But the measure in itself has no story to tell.

Measures used in the experiment. We have opted to compute two different measures: the *accuracy* because it is valid on ordinal problems, and the *mean absolute error* MAE to give an idea of how close the prediction is to the actual label. In the description of the used methods in Section 5.4.2, it is always stated whether the method is monotone or not. To alleviate computations, we did not check out the MSE nor Kendall's tau, this decision was also loosely based on the observation that for the conducted experiments, the MSE behaves rather similar to the MAE, and Kendall's tau to the classification accuracy (see also the experiments presented in Section 7.5.3 (see p. 210)).

5.4.4 Overview of the experimental designs

Artificial data. The first set of experiments we conducted are based on artificially generated monotone samples $\mathcal{S}$. From $\mathcal{S}$, we then sampled (without replacement) the training and test set $\mathcal{S}_{\text{train}}$ and $\mathcal{S}_{\text{test}}$.
Because of the heavy demands on computer power for generating these artificial monotone data sets, we only experimented with relatively small sample sets. It has to be noted that data sets mentioned in the literature share this small scale characteristic since either this kind of data stems from (expensive) surveys and polls, or because firms do not yet keep extensive track of this kind of data.
We consider 5 designs, 3 with monotone data, and 2 with projections of monotone data. The characteristics of these 5 designs can be found in Table 5.5. We use different sizes for the learning sample (shown in the first column of the result tables) to test the effect of the sample size on the performance. The size for the test sample is fixed on 500.
All results collected in the tables (Tables 5.7 and 5.8) are averages (and standard deviations) derived from 40 independent runs. For each row, the result of the best

[6]Just as for example transitivity is a property of a relation. Either the relation is transitive or it is not, it is never transitive to some degree. This is not in contradiction with the different (nested) types of transitivity from fuzzy set theory (the so-called T-transitivity): there the relation is in fact graded, and the methods either do or do not have a certain type of transitivity. The fact that these types can be nested, just means that one type of transitivity necessarily implies the other.

	design 1	design 2	design 3	design 4	design 5
$\|C\|$ (# generated criteria)	4	8	15	8	10
$\|C'\|$ (# used criteria)	4	8	15	4	8
$\|\mathcal{X}_c\|$ (# criterion values)	7	5	5	7	5
$\|\mathcal{L}\|$ (# labels)	7	4	4	7	4
$\|\mathcal{X}\|$ (# possible vectors)	2.401	390.625	1.073.741.824	2.401	390.625

Table 5.5: Characteristics of the artificial data sets.

performing algorithm is put in boldface, the second-best is underlined[7]. The last column indicates the result of a one-sided Wilcoxon signed-rank test [68] between the best and the second best algorithm, indicating whether or not the difference is statistically significant. We use the following symbols: significant using a 99.9% confidence interval $\upuparrows$, significant at the 95% level $\uparrow$, between 95% and 80% $-$, not significant using an 80% confidence interval $\times$.

Data sets from surveys. We also report[8] the results on the four data sets resulting from surveys that were used in [9, 10]. We used 5-fold cross validation and collected the mean results of these 5 runs on each data set. We considered 2 types of 5-fold cross validation, the typical one using 4/5 of the data for learning and the remaining part for validation, and an atypical one where 1/5 of the data is used for learning and 4/5 for testing (to test the effect of different data sizes).
In Appendix 5.B, we mention the background of these surveys. Table 5.6 summarises the characteristics of these data sets, like the number of criteria, the number of labels and the number of available examples for learning and testing. Mind that we did the experiments on the data sets as we received them without any form of pre-processing.

- *Social workers decision* (SWD). This data set includes criterion values of hypothetical cases of child abuse, and the overall risk for the child as judged by experienced social workers.

- *Lecturer Evaluation* (LEV). A data set including criterion values of hypothetical lecturers, and opinions of Business Administration students about their teaching qualities.

- *Employee Selection* (ESL). This data set includes actual criterion values of applicants for an industrial opening, and judgements of recruiting experts about their qualifications for these jobs.

[7]If OSDL and B-OSDL constitute the two best results, we underline the third best result and use it for comparison. Similar for C4.5 and C4.5-pruned.

[8]Many thanks to Arie Ben-David for supplying us these data sets.

- *Employee Rejection/Acceptance* (ERA). The data set includes criteria of hypothetical applicants for a job, and evaluations of Business Administration students regarding their qualifications.

	SWD	LEV	ESL	ERA
$\|C\|$ (#criteria)	10	4	4	4
$\|\mathcal{X}_c\|$ (# criterion values)	2×4; 7×3; 1×2	4×5	2×10; 2×7	4×15
$\|\mathcal{L}\|$ (# labels)	6	5	10	10
$\|\mathcal{X}\|$ (# possible vectors)	69.984	625	4.900	50.625
$\|\mathcal{S}\|$ (# examples)	3.240	3.700	488	5.148

Table 5.6: Characteristics of the survey data sets. (In the row $|\mathcal{X}_c|$, $n \times m$ means there are n criteria with $|\mathcal{X}_c| = m$.)

5.4.5 Results and discussion

Artificial monotone data. For non-stochastic monotone data, the two algorithms OSDL and B-OSDL coincide, so we report only the results of OSDL. From the result depicted in Table 5.7 it is obvious that OSDL outperforms all the other algorithms. It always leads to the lowest mean absolute error, and in general, the improvement is noticeable. Moreover, also for accuracy it delivers the best results except in a few couple of cases, where it ends in a competitive second place.
The disappearance of the good performing algorithm MDT for designs 2 and 3 might seem a bit odd, but this algorithm is simply not able to handle these designs. In most cases, it ran out of memory (which could probably be solved by another implementation), but more importantly, when it did produce a result, it delivered trees with tons of leafs even for problems with small sample size (experimental results can be found in Appendix 3.B (see p. 80)) making it more a black box model than an easy to interpret tree. We will discuss the reason behind this in Appendix 3.A (see p. 80).

Artificial monotone data, projected. Evidently, because of the more difficult learning task, the results can never be as good as for monotone data. Also, the method MDT can no longer be applied since the data sets are no longer monotone. As was the case for the monotone data sets, the OSDL algorithms deliver the best overall results, where B-OSDL seems to be the better trade-off between accuracy and mean absolute error.
It also strikes that, except for design 1, which is a rather small design, the (non-monotone) Naive Bayes method returns good to very good results, both for monotone and non-monotone data.
Besides some common tendencies, some of the results are clearly different when compared to the monotone data sets. In design 4 we see that OLM performs really well, however, not in design 5, nor in the monotone designs. This seems to indicate

design 1

accuracy

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	MDT	OSDL	
100	40.56 ± 3.98	33.15 ± 4.56	30.6 ± 2.28	34.23 ± 3.26	37.11 ± 3.73	36.72 ± 3.31	38.92 ± 3.7	50.68 ± 4.15	**52.29 ± 3.83**	↑
300	53.0 ± 4.34	44.55 ± 4.35	39.71 ± 2.87	40.35 ± 2.92	59.52 ± 3.42	55.29 ± 3.13	60.17 ± 3.5	65.5 ± 3.72	**67.23 ± 3.02**	↑
500	60.4 ± 3.18	51.08 ± 3.69	46.8 ± 2.29	43.42 ± 3.06	69.43 ± 3.1	65.46 ± 2.87	69.61 ± 3.09	72.61 ± 2.28	**73.47 ± 2.47**	↑
700	65.2 ± 3.81	55.52 ± 3.8	51.82 ± 2.54	45.98 ± 3.0	75.64 ± 3.01	71.62 ± 2.79	75.72 ± 2.97	**77.73 ± 3.05**	77.64 ± 2.16	×
1000	70.43 ± 3.25	60.41 ± 4.01	58.62 ± 2.16	47.26 ± 2.78	81.58 ± 2.39	77.62 ± 2.24	81.6 ± 2.39	81.56 ± 2.54	**82.6 ± 2.29**	↑

MAE

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	MDT	OSDL	
100	0.92 ± 0.12	1.21 ± 0.22	1.27 ± 0.08	1.05 ± 0.07	0.97 ± 0.11	1.09 ± 0.1	0.94 ± 0.12	0.64 ± 0.09	**0.53 ± 0.06**	⇈
300	0.63 ± 0.08	0.83 ± 0.13	1.06 ± 0.06	0.9 ± 0.06	0.5 ± 0.06	0.6 ± 0.06	0.5 ± 0.06	0.4 ± 0.05	**0.34 ± 0.03**	⇈
500	0.49 ± 0.06	0.66 ± 0.08	0.88 ± 0.06	0.86 ± 0.05	0.34 ± 0.04	0.43 ± 0.04	0.34 ± 0.04	0.3 ± 0.03	**0.27 ± 0.02**	⇈
700	0.41 ± 0.06	0.58 ± 0.07	0.78 ± 0.05	0.84 ± 0.04	0.27 ± 0.04	0.33 ± 0.04	0.27 ± 0.04	0.24 ± 0.04	**0.23 ± 0.02**	↑
1000	0.34 ± 0.05	0.49 ± 0.07	0.64 ± 0.06	0.82 ± 0.04	0.19 ± 0.03	0.25 ± 0.03	0.19 ± 0.03	0.19 ± 0.03	**0.17 ± 0.02**	⇈

design 2

accuracy

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	
100	40.46 ± 4.9	37.24 ± 4.91	35.22 ± 2.32	**42.21 ± 2.99**	29.13 ± 2.8	30.22 ± 1.36	31.74 ± 2.33	38.26 ± 2.06	↑
300	45.52 ± 5.14	42.4 ± 4.75	38.0 ± 2.63	47.82 ± 3.12	36.74 ± 3.2	35.41 ± 1.83	42.02 ± 3.63	**49.82 ± 3.11**	⇈
500	48.76 ± 4.36	45.7 ± 4.28	39.73 ± 2.29	51.55 ± 2.38	42.88 ± 3.99	38.98 ± 2.09	47.87 ± 4.38	**54.24 ± 2.9**	⇈
700	51.01 ± 4.3	47.56 ± 4.82	41.08 ± 2.4	52.59 ± 3.3	48.8 ± 3.61	42.23 ± 2.57	53.46 ± 4.12	**57.26 ± 3.06**	⇈
1000	53.32 ± 4.45	49.99 ± 4.54	41.99 ± 2.7	54.1 ± 3.13	55.75 ± 4.58	45.64 ± 2.41	59.69 ± 4.37	**61.23 ± 2.97**	⇈

MAE

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	
100	0.81 ± 0.1	0.92 ± 0.11	0.92 ± 0.05	**0.68 ± 0.04**	1.12 ± 0.09	1.22 ± 0.03	1.12 ± 0.08	0.72 ± 0.04	⇈
300	0.72 ± 0.1	0.79 ± 0.1	0.85 ± 0.05	0.62 ± 0.04	0.9 ± 0.08	1.03 ± 0.04	0.82 ± 0.09	**0.54 ± 0.04**	⇈
500	0.66 ± 0.08	0.73 ± 0.09	0.81 ± 0.04	0.6 ± 0.03	0.77 ± 0.08	0.93 ± 0.04	0.69 ± 0.08	**0.48 ± 0.04**	⇈
700	0.62 ± 0.07	0.7 ± 0.09	0.77 ± 0.04	0.59 ± 0.04	0.65 ± 0.06	0.84 ± 0.05	0.58 ± 0.07	**0.44 ± 0.03**	⇈
1000	0.58 ± 0.08	0.64 ± 0.08	0.75 ± 0.05	0.58 ± 0.03	0.54 ± 0.07	0.77 ± 0.04	0.48 ± 0.07	**0.39 ± 0.03**	⇈

design 3

accuracy

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	
500	26.35 ± 2.37	27.54 ± 1.87	25.71 ± 1.89	28.62 ± 2.06	27.38 ± 2.23	26.24 ± 1.09	29.64 ± 2.11	**30.44 ± 2.21**	↑
1000	28.0 ± 2.19	27.89 ± 2.58	26.62 ± 1.65	30.76 ± 2.15	29.32 ± 2.72	26.9 ± 2.06	32.4 ± 2.93	**32.93 ± 2.43**	–
1500	29.76 ± 2.92	29.74 ± 2.87	27.19 ± 1.92	33.01 ± 2.6	31.18 ± 2.69	27.53 ± 1.55	35.77 ± 3.23	**36.23 ± 3.5**	×
2000	29.68 ± 2.93	29.66 ± 2.49	27.16 ± 2.06	34.53 ± 2.12	31.26 ± 2.51	27.9 ± 1.89	**37.13 ± 2.87**	36.86 ± 2.63	–
2500	30.25 ± 2.61	31.04 ± 2.29	27.45 ± 2.34	35.99 ± 2.46	32.52 ± 2.92	28.42 ± 1.75	38.57 ± 3.35	**39.15 ± 3.33**	–
3000	31.08 ± 2.78	32.26 ± 3.02	27.93 ± 2.52	36.08 ± 2.69	33.46 ± 2.27	28.23 ± 2.18	39.74 ± 2.5	**40.9 ± 2.37**	↑

MAE

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	
500	1.21 ± 0.05	1.2 ± 0.06	1.23 ± 0.04	0.95 ± 0.03	1.18 ± 0.05	1.46 ± 0.03	1.3 ± 0.06	**0.88 ± 0.04**	⇈
1000	1.18 ± 0.05	1.18 ± 0.07	1.2 ± 0.04	0.92 ± 0.03	1.14 ± 0.05	1.43 ± 0.05	1.21 ± 0.07	**0.82 ± 0.04**	⇈
1500	1.13 ± 0.07	1.13 ± 0.07	1.17 ± 0.04	0.89 ± 0.03	1.09 ± 0.06	1.42 ± 0.04	1.12 ± 0.08	**0.76 ± 0.05**	⇈
2000	1.12 ± 0.06	1.13 ± 0.06	1.16 ± 0.04	0.87 ± 0.04	1.08 ± 0.05	1.41 ± 0.05	1.06 ± 0.07	**0.74 ± 0.04**	⇈
2500	1.11 ± 0.06	1.1 ± 0.07	1.16 ± 0.05	0.85 ± 0.03	1.05 ± 0.06	1.39 ± 0.03	1.01 ± 0.07	**0.7 ± 0.05**	⇈
3000	1.07 ± 0.06	1.07 ± 0.07	1.15 ± 0.05	0.84 ± 0.03	1.02 ± 0.05	1.39 ± 0.05	0.96 ± 0.06	**0.68 ± 0.04**	⇈

Table 5.7: Accuracy and MAE, monotone designs.

design 4 accuracy

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL	
100	18.29 ± 1.89	17.3 ± 2.49	16.89 ± 1.85	18.55 ± 2.06	21.91 ± 2.04	21.01 ± 2.04	22.69 ± 2.49	21.78 ± 2.62	$\mathbf{22.84 \pm 2.51}$	×
300	20.68 ± 1.99	21.1 ± 2.87	17.95 ± 2.1	21.83 ± 2.19	$\mathbf{23.96 \pm 2.21}$	19.1 ± 2.39	22.57 ± 2.45	21.46 ± 2.5	23.77 ± 2.08	×
500	21.21 ± 2.27	22.85 ± 2.66	18.32 ± 2.1	22.85 ± 2.41	$\mathbf{24.49 \pm 2.53}$	18.78 ± 2.63	21.97 ± 2.64	21.11 ± 3.39	24.4 ± 2.34	×
700	22.12 ± 2.27	22.98 ± 2.82	18.45 ± 1.7	23.74 ± 2.59	24.27 ± 2.28	18.63 ± 2.29	20.61 ± 2.41	20.34 ± 2.78	$\mathbf{24.73 \pm 2.69}$	–
1000	23.45 ± 3.43	23.43 ± 2.92	19.26 ± 2.66	25.18 ± 3.02	24.68 ± 2.91	18.13 ± 2.92	20.46 ± 2.62	20.82 ± 2.97	$\mathbf{25.34 \pm 3.18}$	×

design 4 MAE

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL	
100	1.97 ± 0.13	2.08 ± 0.25	2.02 ± 0.11	$\mathbf{1.6 \pm 0.06}$	1.76 ± 0.14	1.91 ± 0.17	1.82 ± 0.21	1.63 ± 0.12	1.63 ± 0.14	↑
300	1.82 ± 0.1	1.86 ± 0.18	1.99 ± 0.1	$\mathbf{1.49 \pm 0.06}$	1.68 ± 0.14	2.06 ± 0.18	1.91 ± 0.18	1.52 ± 0.09	1.54 ± 0.1	↑
500	1.78 ± 0.15	1.78 ± 0.18	1.9 ± 0.13	$\mathbf{1.45 \pm 0.09}$	1.66 ± 0.16	2.22 ± 0.24	2.0 ± 0.22	1.45 ± 0.1	1.49 ± 0.11	×
700	1.75 ± 0.12	1.77 ± 0.16	1.87 ± 0.1	1.43 ± 0.07	1.66 ± 0.13	2.22 ± 0.22	2.05 ± 0.2	$\mathbf{1.42 \pm 0.08}$	1.45 ± 0.1	×
1000	1.7 ± 0.16	1.72 ± 0.18	1.78 ± 0.14	$\mathbf{1.4 \pm 0.09}$	1.68 ± 0.17	2.26 ± 0.24	2.13 ± 0.22	1.4 ± 0.12	1.42 ± 0.13	×

design 5 accuracy

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL	
100	31.86 ± 3.28	29.86 ± 3.54	29.67 ± 2.4	33.28 ± 2.75	29.66 ± 2.92	31.57 ± 1.54	34.83 ± 2.74	35.39 ± 3.17	$\mathbf{35.41 \pm 3.16}$	–
300	34.97 ± 3.8	35.44 ± 3.93	31.01 ± 2.5	38.72 ± 3.56	34.76 ± 3.08	35.65 ± 2.11	$\mathbf{41.67 \pm 3.28}$	41.53 ± 3.28	41.52 ± 3.3	×
500	36.93 ± 3.08	36.12 ± 3.26	32.44 ± 2.38	40.73 ± 3.16	37.63 ± 3.3	37.67 ± 2.58	$\mathbf{44.53 \pm 3.33}$	44.03 ± 2.93	43.8 ± 2.83	×
700	39.44 ± 4.21	38.38 ± 3.47	32.17 ± 2.57	42.82 ± 2.98	40.51 ± 3.25	40.28 ± 2.36	$\mathbf{46.54 \pm 3.43}$	46.28 ± 3.12	46.25 ± 3.2	–
1000	40.15 ± 4.47	40.37 ± 4.43	33.98 ± 2.27	44.03 ± 3.54	42.98 ± 2.94	41.6 ± 2.63	$\mathbf{48.43 \pm 3.78}$	47.11 ± 3.95	47.14 ± 3.68	↑

design 5 MAE

size	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL	
100	1.06 ± 0.09	1.11 ± 0.12	1.1 ± 0.07	0.86 ± 0.05	1.11 ± 0.08	1.21 ± 0.05	1.04 ± 0.09	$\mathbf{0.81 \pm 0.06}$	$\mathbf{0.81 \pm 0.06}$	⇈
300	0.97 ± 0.08	0.99 ± 0.1	1.05 ± 0.06	0.78 ± 0.05	0.98 ± 0.07	1.02 ± 0.07	0.83 ± 0.08	$\mathbf{0.74 \pm 0.06}$	$\mathbf{0.74 \pm 0.06}$	⇈
500	0.92 ± 0.08	0.96 ± 0.09	1.01 ± 0.06	0.76 ± 0.04	0.91 ± 0.06	0.95 ± 0.05	0.75 ± 0.07	$\mathbf{0.71 \pm 0.06}$	$\mathbf{0.71 \pm 0.05}$	⇈
700	0.88 ± 0.08	0.91 ± 0.07	0.98 ± 0.06	0.74 ± 0.03	0.84 ± 0.06	0.88 ± 0.05	0.71 ± 0.07	$\mathbf{0.68 \pm 0.05}$	$\mathbf{0.68 \pm 0.05}$	⇈
1000	0.85 ± 0.09	0.85 ± 0.09	0.95 ± 0.05	0.73 ± 0.04	0.79 ± 0.05	0.83 ± 0.06	0.68 ± 0.08	$\mathbf{0.66 \pm 0.07}$	$\mathbf{0.66 \pm 0.06}$	↑

Table 5.8: Accuracy and MAE, projected monotone designs.

that OLM is especially suited for difficult monotone learning tasks, where a lot of reversed preferences occur.
Another point worth noticing is that the minimal extension is able to outperform many of the other methods, except in design 4. Even more peculiar at first sight is that in this design, the performance of the minimal (and maximal) extension decreases as more data becomes available. This is however easily explained by noticing that as more examples violate monotonicity, the minimal (resp. maximal) extension becomes more and more extreme, classifying more and more points to the maximal (resp. minimal) label. Indeed, if we reconsider the extreme situation of Table 5.3, then $\lambda_{\min}$ would assign all objects to class 2 and $\lambda_{\max}$ would rank them all as 1.

Data from surveys. These real data sets are not monotone. The results of the 5-fold cross validations are shown in Table 5.9.
The OSDL algorithms continue their good overall performance, and the fact that B-OSDL seemed to be preferred for the artificial data is confirmed in these real-world application domains. However, the differences in performance are much less pronounced.
There is no real winner among the other classifiers, they all comprise a "best" and a "second best" result, although C4.5 and OLM seem to be a bit more successful.
It is remarkable how extremely poor the minimal and maximal extensions perform. Also remark the deterioration in performance of these two classifiers when more data becomes available, just as in design 4.

5.5 Future research

EXPERIMENTS. Although we did some extensive testing, some of the design parameters remained unexplored, like the number of class labels. It would be interesting to run experiments changing only $|\mathcal{L}|$ to gauge its effect on the different measures.
Also, experiments where the second parameter in B-OSDL is tuned to the data would be welcome.

INTERPOLATION. The interpolation technique used at present in the (B-)OSDL algorithm is extremely simple. Incorporating a more sophisticated interpolation, e.g. based on ideas coming from [8] constitutes a promising avenue.

PRE-PROCESSING. As for any instance-based learning algorithm, a good attribute (criterion) selection method can probably improve the results. Possible measures for such a selection scheme could be the OO-measure [48] (see p. 70), or the measure we proposed in [27].

MORE SUBTLE USE OF INFORMATION. The OSDL algorithms merely look at the two dominance regions $[\mathbf{x})$ and $(\mathbf{x}]$, forgetting all about the rest of the data space $\mathcal{X}$, even though there is obviously interesting information to be gathered from other regions. Moreover, all elements in $[\mathbf{x})$, resp. $(\mathbf{x}]$, are treated the same, whereas it might be argued that the vectors closer to the borderlines should be less

data	train	test	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL
SWD	1/5	4/5	55.89 ± 0.89	54.56 ± 0.92	$\mathbf{55.93 \pm 0.92}$	55.91 ± 1.15	55.13 ± 1.35	35.84 ± 3.78	41.23 ± 4.55	55.38 ± 1.13	55.92 ± 0.88
SWD	4/5	1/5	58.46 ± 2.35	58.46 ± 2.56	58.58 ± 2.47	57.62 ± 2.86	58.06 ± 2.59	22.96 ± 2.61	28.8 ± 2.08	58.77 ± 2.55	$\mathbf{59.01 \pm 2.84}$
LEV	1/5	4/5	60.31 ± 0.64	58.64 ± 1.08	59.48 ± 1.3	55.37 ± 1.63	58.85 ± 2.58	27.94 ± 6.3	27.16 ± 7	60.47 ± 0.46	$\mathbf{61.52 \pm 0.73}$
LEV	4/5	1/5	$\mathbf{62.81 \pm 1.55}$	61.86 ± 1.79	62.81 ± 2.95	55.68 ± 1.76	61.92 ± 2.52	14.27 ± 1.15	11.65 ± 3.39	62.41 ± 2.3	62.54 ± 2.28
ESL	1/5	4/5	56.76 ± 3.91	53.53 ± 3.68	59.27 ± 3.14	59.53 ± 2.85	60.3 ± 3.73	54.1 ± 5.71	58.35 ± 2.31	$\mathbf{62.5 \pm 3.09}$	61.83 ± 3.14
ESL	4/5	1/5	66.39 ± 4.01	62.93 ± 4.07	64.55 ± 1.43	64.56 ± 3.3	$\mathbf{69.06 \pm 4.02}$	42.62 ± 4.82	54.94 ± 7.34	66.21 ± 5.01	68.25 ± 2.48
ERA	1/5	4/5	$\mathbf{24.46 \pm 0.65}$	24.37 ± 0.93	23.53 ± 0.28	23.23 ± 0.81	23.63 ± 0.47	11.04 ± 1.36	4.94 ± 1.12	23.82 ± 0.37	23.92 ± 0.37
ERA	4/5	1/5	$\mathbf{26.38 \pm 1.57}$	25.8 ± 1.52	23.8 ± 1.26	25.02 ± 1.8	23.8 ± 1.26	8.61 ± 0.36	2.33 ± 0.31	23.93 ± 1.39	24.11 ± 1.55

data	train	test	C4.5	C4.5 pruned	k-NN	Bayes	OLM	Max	Min	OSDL	B-OSDL
SWD	1/5	4/5	0.47 ± 0.02	0.49 ± 0.01	0.52 ± 0.01	$\mathbf{0.45 \pm 0.01}$	0.47 ± 0.01	0.78 ± 0.07	0.72 ± 0.1	0.47 ± 0.01	0.46 ± 0.01
SWD	4/5	1/5	0.44 ± 0.03	0.44 ± 0.03	0.44 ± 0.03	0.45 ± 0.04	0.43 ± 0.03	1.03 ± 0.03	0.99 ± 0.08	0.43 ± 0.03	$\mathbf{0.42 \pm 0.03}$
LEV	1/5	4/5	0.44 ± 0.01	0.46 ± 0.02	0.45 ± 0.02	0.47 ± 0.01	0.46 ± 0.03	1.07 ± 0.29	1.05 ± 0.23	$0.43 \pm 0.00^{\dagger}$	$\mathbf{0.42 \pm 0.01}$
LEV	4/5	1/5	0.41 ± 0.02	0.42 ± 0.02	0.41 ± 0.02	0.46 ± 0.02	0.41 ± 0.03	1.52 ± 0.07	1.54 ± 0.2	0.41 ± 0.02	$\mathbf{0.40 \pm 0.02}$
ESL	1/5	4/5	0.50 ± 0.04	0.53 ± 0.05	0.5 ± 0.04	0.43 ± 0.02	0.44 ± 0.04	0.53 ± 0.06	0.49 ± 0.01	$\mathbf{0.42 \pm 0.03}$	0.43 ± 0.05
ESL	4/5	1/5	0.37 ± 0.05	0.41 ± 0.06	0.43 ± 0.03	0.36 ± 0.03	$\mathbf{0.33 \pm 0.04}$	0.64 ± 0.07	0.49 ± 0.06	0.37 ± 0.05	0.34 ± 0.04
ERA	1/5	4/5	1.38 ± 0.01	1.39 ± 0.02	1.40 ± 0.03	1.31 ± 0.03	1.27 ± 0.02	2.72 ± 0.21	3.46 ± 0.36	1.26 ± 0.01	$\mathbf{1.26 \pm 0.01}$
ERA	4/5	1/5	1.31 ± 0.05	1.32 ± 0.06	1.33 ± 0.05	1.29 ± 0.06	1.25 ± 0.04	3.12 ± 0.06	4.25 ± 0.18	1.25 ± 0.04	$\mathbf{1.24 \pm 0.05}$

$\dagger$: 0.004

Table 5.9: Accuracy and MAE, data from surveys.

imposing than vectors that are "deep inside" the region. To obtain such generalisations, two paths can be followed:

- use a more involved estimation of the distributions $\hat{F}(\mathbf{x})$ for the $\mathbf{x} \in \mathcal{S}_{\mathcal{X}}$. Any distribution classifier can be chosen for this. This is the simplest method to create a monotone version of any distribution classifier.

- investigate and incorporate the idea of a graded dominance relation, and with it the effect on (graded) transitivity properties (because the OSDL algorithm heavily relies on transitivity).

MONOTONE BAYES. The results from the experiments demonstrated that imposing a monotonicity demand on a classifier will boost its performance in ranking problems (when the data is described by criteria). Because the Naive Bayes algorithm delivered overall good results, it would be worthwhile to investigate how it could be extended towards a monotone version. This could be done as mentioned above, and/or by incorporating monotonicity from the beginning into the method. We did some preliminary explorations on the survey data, using B-OSDL with estimations obtained from the Naive Bayes algorithm, but while they were better than for Naive Bayes alone, they still did not outperform B-OSDL, though coming very close. This suggests that rather the second path should be explored.

OTHER ALGORITHMS. In the real-world application domains, the other algorithms also performed well, so monotone extensions of decision trees (for non-monotone data) and of nearest neighbour algorithms would also be interesting lines of thoughts. In Appendix 5.A, we propose a first line of thought concerning the adaptation of the nearest neighbour algorithm.

Interlude

THE PROJECT

SATURDAY, MAY 18, 2003. *Last week (or was it two weeks ago? time can be so slippery), Bernard asked me to help him write a proposal for a new project. A kind of continuation of my research, combined with his joint work with Hans De Meyer on alternatives for stochastic dominance and its consequences in statistics. Something to do with very strange dice and winning every game when playing with them. Hey, even more abstract than what I'm doing :-)*

Well today, this drizzly evening to be exact, I finally started writing the proposal. It's due this Monday, so – as usual – I must admit I could have began just a tiny bit earlier instead of rushing me in these last minute late night stressy situations. Anyway, the general layout for the proposal is: what's classification, one third of a page, what's the difference with ranking, one third of a page, blahblahblah..., what makes it so difficult to adapt classification algorithms to ranking problems, one third of a page... Hum... Not more than one third of a page? That's though... Let me think..., yeah,... the easiest classification algorithm is surely the Nearest Neighbour one. It's easily explained and it's extremely local, which helps to highlight the difference with the very global monotonicity of the ranking problem. Perfect. What's the next point in the proposal's layout?

Of course, I couldn't just let it be at that. Noooo, not me. I seem to have this compulsive behaviour, this never failing tremendous urge, this always painfully present and uncontrollable aching thirst that leaves me no other option than to dig into any open problem I find myself confronted with (...at least, when it seems solvable in a reasonable amount of time – unfortunately, I'm a lousy estimator). So tonight, it happened again, indeed, I just had to satisfy my yearning and attempt to solve the "Monotone Nearest Neighbour" problem.

In the next chapter, you, my dearest reader (well who would have thought, somebody is actually reading this thesis!), can find the results of my findings during the night of May 18th 2003. It's a rather short chapter, mainly because I really don't have the time anymore to work it out completely, other chapters of this thesis that where planned long before do demand my total devotion at this moment.

SATURDAY, JUNE 29. *Finally, the time has arrived to commence this long-planned journey. Since I conceived it, Bernard repetitively told me a chapter on Quasi Monotone Nearest Neighbours (QM-NN) was really not necessary, that nobody would put up a strange face if it was not included, that no soul would even notice its absence, and some days ago I actually ca-*

pitulated, and – with the full understanding that there was no time left to write it anyway – frankly agreed that he was right. Yet in spite of this, look at me, here I am, investing my precious time into it after all. And why? I'll tell you why. There are two reasons, first of all I think it fills a gap between Chapter 4, the framework of intervals and stochastic dominance, and Chapter 5, the application of stochastic dominance. As you can see, there is a part missing about the application of intervals, and that's QM-NN. Isn't that nice? But a second reason – and I have to admit it's not really a scientific one, not at all, it's rather the kind of ridiculous reason my friends would expect from me – is that I sort of like the previous interlude of May 18 (it was in fact the first one I wrote), and I didn't like the idea of having to leave it out. Well, there you have it, it's out in the blue. So I hope you enjoyed reading the interlude, and I also hope you will enjoy reading the next part of the appendix (as a compromise, I decided to make it part of the appendix rather than a full fledged autonomous chapter).

5.A QM-NN: Quasi Monotone Nearest Neighbour

Here we propose a very simple adaptation of the basic k-Nearest Neighbour algorithm with $k = 1$. Instead of simply returning the label of the nearest label, we will make sure it behaves monotonically w.r.t. to the given learning samples by first identifying for the object the adequate interval of class labels to which the label of the object should belong.

QUASI-MONOTONE. **Notions and conventions.** A function $f : (X, \leq_X) \to (Y, \leq_Y)$ is called **quasi-monotone** [74] w.r.t. a subset $S \subseteq X$ if all elements $x \in X$ are monotone w.r.t. all elements $s \in S$, i.e. for all couples $(x, s) \in X \times S$ we have $x \leq_X s$ implies $f(x) \leq_Y f(s)$ and $s \leq_X x$ implies $f(s) \leq_Y f(x)$. Clearly, f can only be monotone w.r.t. S if $f_{|S}$ is monotone itself.

5.A.1 Interval representation

We start with an adaptation of Theorem 4.6.3 (see p. 104) in the context of supervised learning. It handles the problem of data fitting.

Theorem 5.A.1. *Let $(d, \leq_{\mathcal{S}})$ be a ranking in $\mathcal{S}$, and let C be a set of criteria. Now define*

$$\boxed{\tilde{\lambda}(\mathbf{x}) := \begin{cases} [\lambda_{\max}, \lambda_{min}] & \text{, if } \lambda_{\max} \leq_{\mathcal{L}} \lambda_{min} \quad \text{(monotone case)} \\ [\lambda_{\min}, \lambda_{max}] & \text{, if } \lambda_{\min} \leq_{\mathcal{L}} \lambda_{max} \quad \text{(non-monotone case)} \end{cases}}$$

where $\lambda_{\min}$ and λ_{max} are the minimal and maximal extension. We have that the representation $(\tilde{\lambda}, \leq^{[2]})$ is non-decreasing, where $\leq^{[2]}$ is the order on

$$\mathcal{L}^{[2]} = \{[r, s] \mid (r, s) \in \mathcal{L}^2 \wedge r \leq_{\mathcal{L}}\}$$

defined as

$$[r_1, r_2] \leq^{[2]} [s_1, s_2] \iff ((r_1 \leq_{\mathcal{L}} s_1) \wedge (r_2 \leq_{\mathcal{L}} s_2)) \ .$$

5.A.2 QM-NN

Here we present how the interval representation can help into remolding a Nearest Neighbour (NN) algorithm into a *quasi-monotone* (see p. 56) variant, on the condition that the learning sample is monotone. If the learning sample is not monotone, QM-NN no longer has the monotonicity property, the only thing that can be said is that it takes into account the monotonicity into its calculations to enhance performance.

The construction of the data base is exactly the same as for the NN algorithm. For any $\mathbf{x} \in \mathcal{X}$, $\lambda_{\text{NN}}(\mathbf{x})$ is the label produced by NN. The labelling of a new instance $\mathbf{x} \in \mathcal{X}$ by QM-NN is done as follows:

If the learning sample is monotone, then Theorem 5.A.1 assures that the label is chosen within an interval that guarantees monotonicity w.r.t. to the learning samples $\mathcal{S}$. This procedure results by definition in a quasi-monotone labelling w.r.t. $\mathcal{S}$.

Algorithm 5.3 : Label, label a new instance

```
$\mathbf{x} \leftarrow$ new instance to classify;
$\ell \leftarrow \min\{\lambda_{\min}(\mathbf{x}), \lambda_{\max}(\mathbf{x})\}$;
$r \leftarrow \max\{\lambda_{\min}(\mathbf{x}), \lambda_{\max}(\mathbf{x})\}$;
if $\ell = r$ then
   return $\ell$;
else
   $v \leftarrow \lambda_{\text{NN}}(\mathbf{x})$;
   if $v < \ell$ then
      return $\ell$;
   else if $v > r$ then
      return $r$;
   else
      return $v$;
   end if
end if
```

5.A.3 Experiments.

We have conducted a few experiments in the fashion of Section 5.4. We are mainly interested in the comparison with the Nearest Neighbour algorithm, but will also report some other algorithms.

Results and discussion. The results all convey the same message, so we only display one design, using 10 criteria, where each criterion can take 5 values, and there are 4 labels. The data set is monotone. Figure 5.4 shows the mean results after 40 independent runs.

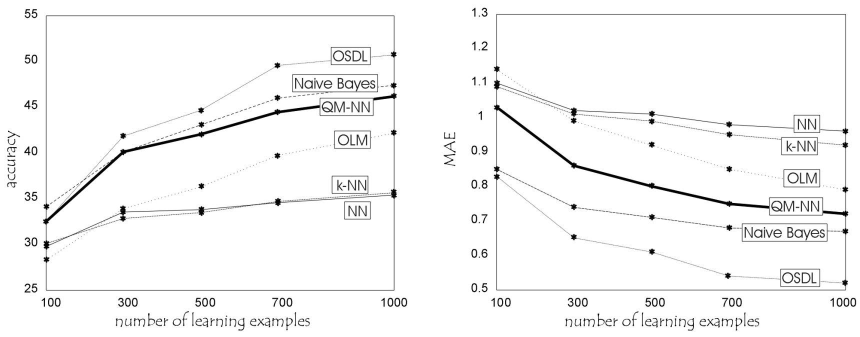


Figure 5.4: Accuracy and MAE, monotone design.

Clearly, incorporating monotonicity requirements into the NN method leads to an obvious gain in performance. However, this simple scheme is still not able to top the Naive Bayes, and is even a longer way from beating OSDL.

5.B Background of the surveys.

We take over the texts from [9] documenting these data sets.

Social Workers Decisions. Monica Shapira and Rami Benbenishty [102] of the [Paul Baerwald] School of Social Work at the Hebrew University of Jerusalem conduct ongoing research regarding decision making processes of social workers [their research is still going on [12, 103]]. In one of their experiments they presented experienced social workers with simulated cases of abused children, asking the experts to judge the risk for the child and to suggest appropriate intervention. Risk judgements were measured on a five point ordinal scale.
Social workers are known to be quite reluctant to interfere with an abused or neglected child's family structure, unless they have solid reasons for doing so. They typically prefer to try and solve the problem within the family. This is the reason why they were found empirically to apply strict decision making strategies [102]. The inputs for the decision were attributes such as the mother's relation toward the child, parental cooperation, severity of signs of abuse, etc. There were ten such attributes, all of which were ordinal. The opinions were given by twenty-nine qualified social workers. A file with examples regarding 3240 rehabilitation program recommendations was used.

Lecturer Evaluation. This data set was taken from an experiment aimed at comparing self-declared problem-solving strategies and actual judgements. Sixty-three undergraduate students were presented with randomly generated profiles of hypothetical lecturers. The students were asked to evaluate each lecturer according to his/her ability to capture student's interest, in achieving appropriate class participation, teaching analytical tools, etc.

Employee Selection. Just as different jobs require different skills, testing candidate's qualifications varies with the type of position, its level, and the resources allocated to the testing procedure. Selected candidates for certain positions are sometimes sent to consulting firms that specialise in evaluating their qualifications trough psychometric tests and interviews. The resulting evaluation serves as an input for the decision of which applicants best fit the positions. The experience gained by a consulting firm influences the method it uses.
In order to evaluate candidates for some industrial manufacturing position, a leading Israeli recruiting firm uses a hierarchical model. The output is a score, with ten possible ordinal values, that predict the candidate's qualification to successfully fill the position. In our data set, there were four top level attributes on which the score was based: working style, writing fluency, ability to fit in the organisation, and other qualifications. Each top-level attribute also had a score with ten possible ordinal values. The score of a top level attribute was determined according to lower level attributes. Working style, for example, was determined by determination, flexibility, curiosity, pragmatism, and openness. Although each sub-model for

determining a lower level attribute was also ordinal, in this experiment only the upper level decision were used. Anonymous actual examples of 488 applicants for certain closely related openings were available.

Employee Rejection/Acceptance. The purpose of this research, conducted by Yoav Ganzach of the Hebrew University of Jerusalem, was to evaluate the degree on non-linearity in decision making strategies, and to evaluate the effects of problem presentation on decision makers. As part of the experiment, 115 undergraduate business administration students were given random profiles of hypothetical candidates for a job. The subjects were told they were personnel managers responsible for hiring employees for a managerial position. they were asked to indicate to what degree (on a seven point scale) they would accept or reject each candidate according to four attributes: maturity, motivation, academic achievement, and interpersonal communication. Each attribute had seven possible ordinal values.
The problem that was presented before the students was ordinal. The cover story implied strong involvement, and the subjects were found to apply strict judgements [43].

Interlude

ABOUT JOY AND AGONY

Every Ph.D. student is familiar with the regular depressive time lapses that are adequately called Ph.D. blues*. Nobody escapes it (except maybe the ones that really do not care about their Ph.D. - and I know only of one such a person whom I know will be smiling broadly with twinkling eyes when reading this), it hits you at least once during your Ph.D., sometimes weakly, sometimes harsh, and for quite some people it doesn't fade away until they take refuge in the ultimate dramatic cure of forsaking their thesis altogether. But every bad comes with a good, and this is no exception to this rule. As a compensation for the blues periods, we also get funky periods, when everything is shiny and bright and beautiful and this without being in love! These are the moments when two pieces finally fit together or an experiment leads to better results than even expected, when some mysterious haze that clouded your mind suddenly dissipates under the warmth and light of a new insight.*

MONDAY, MAY 19, 2003. *During a long time, I thought that Chapter 6 was already finished... until that dreadful moment in April (dearest reader, I beg your indulgence for the non-chronological account of my story, I will tell all about that dreadful moment in the Interlude preceding Chapter 7). I wrongfully thought I just had to copy paste the article I wrote for Intelligent Systems and do some minor adjustments. Today, I know I need to incorporate the property of transitivity into my definition of partial dominance, and I even know how it can be done, namely be simply considering the transitive closure of my earlier definition. So, all things considered, it should still not be too much work. Let's start writing and get it over with!*

TUESDAY, MAY 20. *Oh boy, I am afraid I proved – again – to be a lousy estimator. Again, I underestimated my animal-like urge too understand the why, and, as always, I now realise that I will not be satisfied by simply describing the how (as I – stupid me – still erroneously thought yesterday). The only problem is that, at this very moment, I only have a strong intuition about the why, a feeling that all but breaks the tender film between hunch and knowledge, an understanding that balances on the edge of chaotic intuition and ordered formalisation. In the end, it just means that nobody but myself will appreciate it if I would write it down like that. So here I am, like a proper fidget, nudging and budging and stretching my mind to come up with a decent formal way of confiding my intuition to paper.*

After a lot of huffing and puffing, I finally managed to blow away the mysterious clouds that blocked my view, and I became the proud father of Section 6.2.3, "Dominance revisited". I'm really feeling good!! Tomorrow, I will wrap this chapter up!

WEDNESDAY, MAY 21. *Or not.*

Today was a whirlpool of emotions. I've seen heights, I've seen depths. What started as a nice and easy mountain walk on a pleasant summer afternoon, soon turned into a difficult and tricky ascension. After falling, I crawled up again, reaching to the summit, only to find that the weather had turned midway, and to get myself lost in the ever thickening mists on the flanks of what had become mount Doom. The joy and agony, blissful happiness and endless sorrow that are so typical during a Ph.D. but are normally stretched out over wide periods of time, now succeeded each other in a daredevil raging canter. There was scarce time to breath in between. One moment, I believed I had the final solution, the next, it was scattered into bits and pieces again. I fixed it, only to find it break apart at another spot. And I don't have the luxury of time anymore, to leave it to settle all by itself. I need that answer now, today!

THE FOLLOWING DAYS. *At last, I managed to crest this high rising mountain and was rewarded with a splendid view on the tricky slopes of slide debris going up sheer walls of rock, on the deeply cut valleys at their base, and on the open hillsides and wooded vales scattered around in the far distant.*

Chapter 6

A framework for ranking: relational granulation

6.1 Introduction

This is the third and final chapter about the framework for rankings. Here the previous two (namely Chapters 2 and 4) are blended together, and served with some additional spices.

Before, we have mixed and kneaded the basic ingredients of Section 2.2 into Section 2.3. In the present chapter, we will mimic this recipe and mould Chapter 4, the counterpart of Section 2.2 for rankings, into a more general and more flexible form.

To do so, we first need to understand better our basic ingredients. The elementary syntax for classification used in Section 2.2 was rather matter-of-course. However, we saw that for ranking, the syntax was richer and additionally seasoned by some semantical flavours. Therefore, we start in Section 6.2 by dissecting and scrutinising the *dominance relation* and the associated *principle of dominance preservation*, which lead to monotonicity requirement in rankings (Section 4.5 (see p. 97)). These two notions are then opened up respectively in Sections 6.3 and 6.4. Finally, with Section 6.5, we complete this chapter by serving some of the more technical aspects concerning the representation of rankings.

Preliminaries about ranking. We quickly refresh the main ideas and concepts of the supervised ranking problem. A **classification** (see p. 29) is a function $\lambda : \Omega \to \mathcal{L}$, assigning class labels (e.g. "red", "blue", "yellow") to a set of objects. A (complete) **ranking** (see p. 90) is a classification λ where the labels are linearly ordered by the (complete) order $\leq_{\mathcal{L}}$ (a reflexive, antisymmetric and transitive relation on $labels$) and this ordering reflects a preference between the classes (e.g. "Bad", "Satisfactory", "Good"). We denote a ranking as $(\lambda, \leq_{\mathcal{L}})$.

CLASSIFICATION λ.

RANKING $(\lambda, \leq_{\mathcal{L}})$.

In the context of classification, the object space Ω is structured by describing the objects on the basis of a fixed set Q of **attributes** (see p. 30) $q : \Omega \to \mathcal{X}_q$. Objects are then represented by vectors in $\mathcal{X} = \prod_{q \in Q} \mathcal{X}_q$. In the context of ranking, it is usual to consider criteria instead of attributes. A **criterion** [97] (see p. 98) is defined as a mapping $c : \Omega \to (\mathcal{X}_c, \geq_c)$, where $\geq_c$ is a complete order on $\mathcal{X}_c$, such that it appears meaningful to compare two objects a and b, according to a particular point of view, on the sole basis of their evaluations $c(a)$ and $c(b)$. We will only consider so-called *true criteria* [20]: a is preferred to b according to criterion c if $c(a) >_c c(b)$.

ATTRIBUTES.

CRITERION.

6.2 The principle of dominance preservation

Notions and conventions A **preorder** (see p. 88) is a reflexive and transitive relation. A **weak preference relation** [118] (see p. 88) S is a reflexive relation where the expression aSb stands for "a is at least as good as b". In this chapter, we will use the notation $\succcurlyeq_S$ instead of S.

PREORDER.

WEAK PREFERENCE RELATION $\succcurlyeq_S$.

A weak preference relation can be decomposed into (and is totally defined by) three mutually exclusive relations: an asymmetric *strict preference relation* $\prec_S$ ($a \prec_S b$ if $a \preccurlyeq_S b$ and not $b \preccurlyeq_S a$), a reflexive and symmetric *indifference relation* $\sim_S$ ($a \sim_S b$ if $a \preccurlyeq_S b$ and $b \preccurlyeq_S a$), and an irreflexive and symmetric *incomparability relation* $\|_S$ ($a \|_S b$ if not $a \preccurlyeq_S b$ and not $b \preccurlyeq_S a$).

$\prec_S, \sim_S, \|_S$

6.2.1 An example

Example (part 1). We start with the small introductory example that was presented in Section 1.2.2 (see p. 6), the data are shown again in Table 6.1, but for an elaboration on this candidate evaluation example, you will need to thumb back to Chapter 1. We assume familiarity with the basics of decision trees (see also Appendix 1.A, p. 14).

	c_1	c_2	c_3	λ
a_1	$-$	$-$	$+$	B
a_2	$+$	$-$	$-$	M
a_3	$-$	$+$	$+$	G
a_4	$+$	$+$	$-$	M

Table 6.1: Evaluations of candidates.

If we would run a classification tree algorithm on this problem, we would end up with one of the two trees depicted in Figure 6.1. If we choose

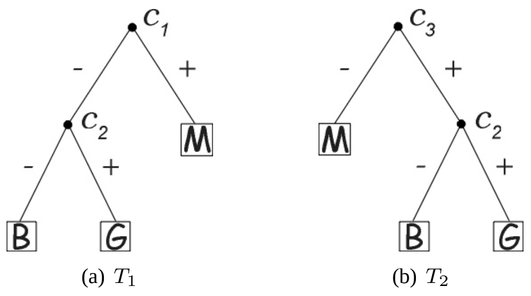


Figure 6.1: Classification trees for candidate evaluation.

tree T_1, for instance, it turns out that the best possible candidate, with evaluations $(+,+,+)$, is evaluated as Moderate. However, another person having evaluations $(-,+,-)$ ends up in the class labelled Good. This is in contradiction with the basic *principle of dominance preservation*, roughly stating that an object a with (partial) evaluations at most as good as the (partial) evaluations of an object b, should have a global evaluation that is also at most as good.

6.2.2 The principle

The principle of dominance preservation. We start by repeating the definition of the *dominance relation* (see p. 98).

Definition 6.2.1 (see [97])

The **dominance relation** $\lhd$ on Ω w.r.t. a set of true criteria C is defined by

$$a \lhd b \iff \begin{cases} (\forall c \in C)(c(a) \leq_c c(b)) \\ (\exists c \in C)(c(a) <_c c(b)) \end{cases}$$

for any $a, b \in \Omega$. It is said that a *is dominated by* b. If only the first condition is fulfilled, i.e. $(\forall c \in C)(c(a) \leq_c c(b))$, we say that a **is weakly dominated by** b and we write $a \unlhd b$.

PRINCIPLE OF DOMINANCE PRESERVATION.

The **principle of dominance preservation** can now be formulated as

$$a \lhd b \implies \lambda(a) \leq_{\mathcal{L}} \lambda(b)\,. \tag{6.2.1}$$

If this principle is violated, we speak of *reversed preference* (see p. 100) between the ranking $(\lambda, \leq_{\mathcal{L}})$ and the set of criteria C: there exist objects $a, b \in \Omega$ such that

$$a \lhd b \quad \text{and} \quad \lambda(a) \not\leq_{\mathcal{L}} \lambda(b)\,.$$

We remark that the principle of dominance preservation implicitly demands that the relation on its left hand side is transitive (see Section 6.2.3). In this case, $\lhd$ is transitive, so there is no problem.

6.2.3 Dominance revisited

Introduction. In this section, we have a closer look at the dominance relation. While dominance seems a very simple and straightforward idea, it has quite a few tricky catches well hidden below its serene surfaces. As soon as you try to meddle with its most basic definition, problems start emerging from the depths and threaten to turn the serene surface into a boiling chaos. We will not go too deeply into the subject, and will only dig into the problems that arise in the context of the supervised learning of a ranking.

The usual definition. If we encounter the (weak) dominance relation in the literature, it is always defined as in Definition 6.2.1. And it always goes hand in hand with the following basic principle from MCDA (Multi-Criteria Decision Aid) [97]

$$a \unlhd b \Rightarrow a \preccurlyeq_S b\,, \tag{6.2.2}$$

where $\preccurlyeq_S$ is the weak preference relation on Ω that is sought after during the decision process.

A different view. First we write (6.2.2) in a more general form where the dominance relation is replaced by some weak preference relation $\preccurlyeq_D$. We obtain

$$a \preccurlyeq_D b \Rightarrow a \preccurlyeq_S b\,. \tag{\#}$$

We know that the weak dominance relation fulfills property (#). Let us now turn the world upside down, and instead of saying that some weak preference relation $\preccurlyeq_D$ satisfies (#), we see this property as defining some weak preference relation $\preccurlyeq_D$, which we could interpret as a more general version of the weak dominance relation. We mean the following: the underlying idea of (#) becomes the question: if we know the evaluations of the objects on the different criteria (we have no additional information about any relationships between criteria whatsoever), in which cases can we say that $a \preccurlyeq_S b$? These cases are then captured by the generalised dominance relation, i.e. $a \preccurlyeq_D b$ (a is weakly dominated by b) if and only if we can say, solely based on the information of $c(a)$, $c(b)$ and the derived information about $\preccurlyeq_{S_c}$ and $\sim_{S_c}$ on a and b for all criteria $c \in C$, that $a \preccurlyeq_S b$. We then could define $a \prec_D b$ (a is dominated by b) by $a \preccurlyeq_D b$ and not $b \preccurlyeq_D a$.
Another view could be even less restrictive in the sense that information other than only the evaluations of the objects on the different criteria might also be considered. We call the resulting relation an **integrated dominance relation**. The more information is taken into consideration, the more the integrated dominance relation matches $\preccurlyeq_S$. If all available information is considered, the integrated dominance relation equals $\preccurlyeq_S$ (in the end, $a \preccurlyeq_S b$ just means that a is weakly dominated by b if all available information is taken into account). In that sense, $\trianglelefteq$ is the least integrated dominance relation, the most objective one, i.e. for all integrated dominance relations $\preccurlyeq_D$ we must have $a \trianglelefteq b$ implies $a \preccurlyeq_D b$. The *decision-maker-dominance* used in ARGUS [34, 35] can be catalogued as an integrated dominance relation, somewhere in between $\trianglelefteq$ and the final $\preccurlyeq_S$. Also the so-called *butterflies* in [37] are examples of integrated dominance relations.

INTEGRATED DOMINANCE RELATION.

A valid extension? A plausible and intuitive definition in line of the first more restricted view would be

$$a \prec_D b \iff \begin{cases} (\forall c \in C)(a \preccurlyeq_{S_c} b) \\ (\exists c \in C)(a \prec_{S_c} b) \end{cases}$$

and

$$a \preccurlyeq_D b \iff (\forall c \in C)(a \preccurlyeq_{S_c} b)\,.$$

For *true criteria*, we just find that $\preccurlyeq_D$ coincides with $\trianglelefteq$. However, we do not dare to put this forward as a general definition without a serious investigation of all its consequences: what are the effects of this definition together with the principle (#) on $\preccurlyeq_S$? This is no problem if all the $\preccurlyeq_{S_c}$ behave extremely well as in the case of true criteria, but what if some $\preccurlyeq_{S_c}$ show a more exotic behaviour?

An example of "more exotic behaviour". Assume all criteria $c \in C$ incorporate some indifference threshold q_c, i.e.

$$a \sim_{S_c} b \iff |c(a) - c(b)| \leq_c q_c\,.$$

Now let $a, b \in \Omega$ with $c(a) = c(b) + q_c$ for all $c \in C$. This means that $(\forall c \in C)(a \sim_{S_c} b)$, and consequently, $a \succcurlyeq_D b$, which leads in turn to $a \succcurlyeq_S b$. Analogously, we find $b \succcurlyeq_S a$, whence $a \sim_S b$. Still, if put before the choice between a and b, the human mind tends to prefer a just because $a \triangleright b$. Can we simply construct $\succcurlyeq_S$ and afterwards break ties using $\triangleleft$, or does such a two-step procedure produce undesired side-effects (direct, e.g. rank reversal, or indirect, e.g. via the axiom of independence of irrelevant alternatives)? What if the behaviour is even more exotic, for example, $\sim_{S_c}$ and $\prec_{S_c}$ are no longer transitive?

AXIOM OF INDEPENDENCE OF IRRELEVANT ALTERNATIVES.

A weak preference relation S on Ω satisfies the **axiom of independence of irrelevant alternatives** [4] if the relations between two objects (alternatives) a and b are not influenced by the pairwise relations between a, b and a third alternative e. This means for example that your attitude towards 2 political parties (e.g. the "AB&C" and the cartel "D.E.F-Lively") should not be influenced by the existence (or non-existence) of some third party like "Vote4Me". It is clear that any method that violates this axiom can be criticised as being not robust.

Dominance and transitivity. And with this, we come to the problem that started our musing about the dominance relation. We start with the premisse that the integrated dominance relation is not transitive. What are the consequences together with the rule (#)? Let $\Omega = \{a, b, e\}$, with $\neg(a \preccurlyeq_D b)$. Now take into consideration the independent third party e, with $a \preccurlyeq_D e$ and $e \preccurlyeq_D b$ (such a situation is conceivable because of the premisse that $\preccurlyeq_D$ is not transitive). The rule (#) implies $a \preccurlyeq_S e$ and $e \preccurlyeq_S b$. It is time to add another premisse: we want $\preccurlyeq_S$ to be transitive. In that case, we obtain $a \preccurlyeq_S b$. This means that, with these given premisses, the configuration $b \prec_S a$ becomes impossible because of an independent third party!

Dominance, transitivity and supervised learning. In the supervised learning problem, we always start with only a subset $\mathcal{S}_\Omega$ (the objects that will be used for learning) of the object space Ω. This implies we have to be very careful concerning the axiom of independence of irrelevant alternatives, since every object in $\Omega \setminus \mathcal{S}_\Omega$ becomes an irrelevant alternative.
Let us start again from the premisses that the weak preference relation $\preccurlyeq_S$ is transitive, while the dominance relation is not. As should be clear by now, the axiom of independent third parties is violated in most cases. The obvious question is: Is there a way out of this impasse? The answer is yes: just replace the non-transitive dominance relation by its transitive closure. When closing a reflexive relation R in a transitive way, the resulting transitive relation $\bar{R}$ becomes a *preorder* (see p. 88) with (interpreting $\bar{R}$ as a weak preference relation $\preccurlyeq_{\bar{R}}$)

$$\begin{aligned} a \sim_{\bar{R}} b &\iff (a \preccurlyeq_{\bar{R}} b) \wedge (b \preccurlyeq_{\bar{R}} a), \qquad \text{and} \\ a \prec_{\bar{R}} b &\iff (a \preccurlyeq_{\bar{R}} b) \wedge \neg(b \preccurlyeq_{\bar{R}} a)\,, \end{aligned}$$

meaning that all cycles that where introduced during the transitive closure, result in series of indifferent objects (if $a \preccurlyeq_{\bar{R}} b \preccurlyeq_{\bar{R}} c \preccurlyeq_{\bar{R}} a$, then transitivity implies $a \preccurlyeq_{\bar{R}} b$, $b \preccurlyeq_{\bar{R}} a$, and hence $a \sim_{\bar{R}} b$; likewise $b \sim_{\bar{R}} c$ and $a \sim_{\bar{R}} c$). When applied to a non-transitive dominance relation, this indifference is propagated to $\preccurlyeq_S$ via the rule (#).

This new transitively closed integrated dominance relation already incorporates all possible irrelevant alternatives. This is a quite drastic, but totally justifiable approach. Reconsider the political party example. Closing the integrated dominance relation just means that you have considered the existence of all possible other parties *before* fixing your attitude towards the two parties "AB&C" and "D.E.F-Lively". As a result, the existence (or non-existence) of a party like "Vote4Me" will not influence your attitude anymore since you already incorporated its *possible* existence in your attitude towards the other two parties.

A small remark is in place. The axiom of independence of irrelevant alternatives is only violated because we added some additional information about the properties of S, namely that it is transitive. This means that the transitively closed integrated dominance relation becomes even more integrated.

The principle of dominance preservation. We will now relate this to the principle of dominance preservation (6.2.1). The principle

$$a \triangleleft b \Rightarrow a \preccurlyeq_S b\,,$$

is just a small variant of the principle (6.2.2). In the case of a ranking, $a \succcurlyeq_S b$ corresponds to $\lambda(a) \geq_{\mathcal{L}} \lambda(b)$, whence we obtain the principle of dominance preservation

$$a \triangleleft b \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)\,.$$

Moreover, since $\leq_{\mathcal{L}}$ is transitive, the previous discussion becomes very actual in this context. The use of $\triangleleft$ does not lead to any problems because it is transitive. However, if we try to generalise the dominance relation into a partial dominance relation, with an associated principle of partial dominance, we must be careful and verify transitivity.

6.3 The local dominance relation

Notions and conventions. Let $\mathcal{X} = \prod_{c\in C} \mathcal{X}_c$ for some fixed set of criteria $C = \{c_i \mid i \in N\}$, with $N = \{1, \ldots, n\}$. As usual, we use the conventions $a, b \in \Omega$, $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, with $\mathbf{x} = (x_1, \ldots, x_n)$, and if $a \in \Omega$, then $\mathbf{a} \in \mathcal{X}$ with $a_i = c_i(a)$. The product order $\leq_{\mathcal{X}}$ is defined as $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ if $(\forall i \in N)(x_i \leq_{c_i} y_i)$.

Any subspace $\mathcal{X}_I = \prod_{i\in I} \mathcal{X}_{c_i}$ with $I \in N$ is called a *grid* (see p. 18). The associated product order is denoted by $\leq_{\mathcal{X}_I}$. A *partition* (see p. 33) Π of $\mathcal{X}$ is a set of non-empty, pairwise disjoint subsets π of $\mathcal{X}$ such that $\bigcup_{\pi\in\Pi} \pi = \mathcal{X}$. The elements π of a partition Π are called *blocks*. The unique block containing $\mathbf{x} \in \mathcal{X}$ is denoted by $\Pi(\mathbf{x})$.

6.3.1 Partial knowledge, grids

Example (part 2). If we look at Table 6.1, we see that there is no violation against the principle of dominance preservation. Nevertheless, careless induction of the ranking λ may lead to (partial) reversed preferences. Let us have a closer look at how this can happen in the context of decision trees. If we analyse the possibilities for the first split, we notice that only the second option has some kind of monotone behaviour as can be expected from a true criterion (see Figure 6.2). More specifi-

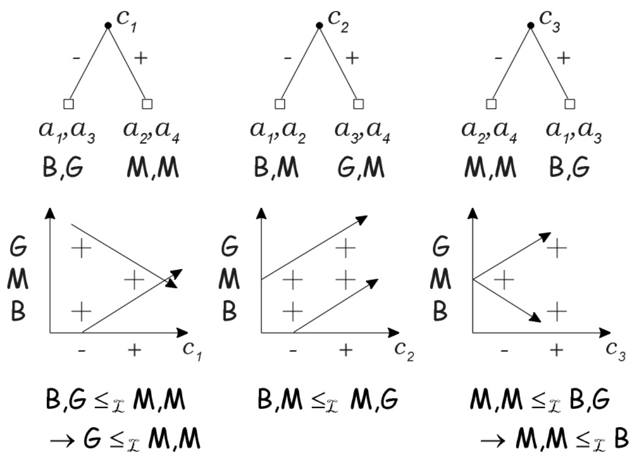

c_1
-
+
a_1,a_3
a_2,a_4
B,G
M,M
c_2
a_1,a_2
a_3,a_4
B,M
G,M
c_3
a_2,a_4
a_1,a_3
M,M
B,G
G
M
B
c_1
c_2
c_3
B,G $\leq_{\mathcal{I}}$ M,M
$\rightarrow$ G $\leq_{\mathcal{I}}$ M,M
B,M $\leq_{\mathcal{I}}$ M,G
M,M $\leq_{\mathcal{I}}$ B,G
$\rightarrow$ M,M $\leq_{\mathcal{I}}$ B

Figure 6.2: Three possible first splits.

cally, we could say that the second option preserves *partial* dominance in the sense that $c_1(a) <_{c_1} c_1(b) \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)$. The split based on c_1, however, results in $c_1(a_3) <_{c_1} c_1(a_2) = c_1(a_4)$, while $\lambda(a_3) = \mathsf{G} \not\leq_{\mathcal{L}} \lambda(a_2) = \lambda(a_4) = \mathsf{M}$. A similar counter-intuitive situation occurs for the split based on c_3.

Grids. Consider the following situation, which is typical for the rough set methodology (see Appendix 1.B, p. 17): assume we only know the values of the objects from Ω on the subset of criteria $C_I = \{c_i \mid i \in I\} \subseteq C = \{c_i \mid i \in N\}$ for some subset $I \subseteq N$. This amounts to saying that for all remaining criteria c, $c(a)$ is an unknown value, and this for all $a \in \Omega$. In other words, instead of working in $\mathcal{X}$, we are now working in the grid $\mathcal{X}_I$. Obviously, it is straightforward to restrict the definition of the dominance relation to the subspace $\mathcal{X}_I$. This is exactly how in [50, 53] the *(weak) partial dominance relation* $\trianglelefteq_I$ on Ω w.r.t. a set of true criteria $C = \{c_i \mid i \in N\}$ and a subset $I \subseteq N$ is defined:

$$a \trianglelefteq_I b \iff (\forall c \in C_I)(c(a) \leq_c c(b)) \,, \tag{$*$}$$

for any $a, b \in \Omega$. Based on this definition, the principle of partial dominance preservation becomes

$$a \triangleleft_I b \implies \lambda(a) \leq_{\mathcal{L}} \lambda(b) \,.$$

It can simply be interpreted as the principle of dominance preservation making abstraction of the criteria that are not under consideration. Remark that this principle of partial dominance preservation is meaningful because $\lhd_I$ is transitive (see Section 6.2.3, p. 154).

6.3.2 Partial knowledge, partitions

Introduction. More generally, we may be in the situation where we only know that an object belongs to some block in a partition. This typically happens in partition-based methods (see Section 2.3, p. 33), in particular tree-based methods (see Appendix 1.A, p. 14). Look for example at Figure 6.3. If we know that an object a falls

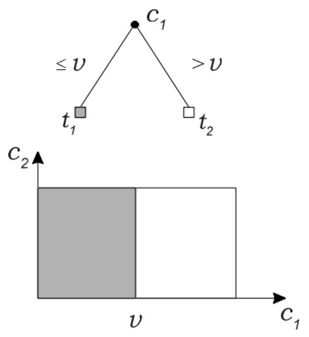


Figure 6.3: A binary tree with the induced partition.

into the leaf t_2, i.e. we just know that $c_1(a) \leq_{c_1} v$ and $c_2(a) \geq_{c_2} w_1$, and if another object b ends up in t_3, i.e. we know that $c_1(b) \geq_{c_1} v$ and $c_2(b) \leq_{c_2} w_2$, how can we compare a and b? If the partition is induced by the grid $\mathcal{X}_I \subseteq \mathcal{X}$, we are in the special case dealt with in the previous paragraph.
It is possible to define the relation $\trianglelefteq_I$ indirectly via the blocks of the partition Π_I induced by the grid $\mathcal{X}_I$. First remark that for all elements $\mathbf{x}, \mathbf{y}$ in the same block $\pi \in \Pi_I$ it holds that $(\forall i \in I)(x_i = y_i)$. So, if for some particular $a, b \in \Omega$ with $\mathbf{a} \in \pi$ and $\mathbf{b} \in \pi'$ it holds that $a \trianglelefteq_I b$ (or equivalently $(\forall i \in I)(a_i \leq_{c_i} b_i)$), then we have immediately that for all $a, b \in \Omega$

$$(\mathbf{a} \in \pi) \wedge (\mathbf{b} \in \pi') \Rightarrow (a \trianglelefteq_I b)\,.$$

This implies that the following definition is meaningful: for all $\pi, \pi' \in \Pi_I$, define

$$\begin{aligned} \pi \trianglelefteq_I \pi' &\iff (\exists \mathbf{x} \in \pi)(\exists \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}_I} \mathbf{y}) \qquad (**)\\ &\iff (\forall \mathbf{x} \in \pi)(\exists \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}_I} \mathbf{y})\\ &\iff (\exists \mathbf{x} \in \pi)(\forall \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}_I} \mathbf{y})\\ &\iff (\forall \mathbf{x} \in \pi)(\forall \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}_I} \mathbf{y})\,, \end{aligned}$$

where $\mathbf{x} \leq_{\mathcal{X}_I} \mathbf{y}$ means $(\forall i \in I)(x_i \leq_{c_i} y_i)$. We can now define $\trianglelefteq_I$ on Ω via $(**)$ as follows: let $a, b \in \Omega$ with $\Pi_I(a) = \pi$ and $\Pi_I(a) = \pi'$, then

$$a \trianglelefteq_I b \iff \pi \trianglelefteq_I \pi' \,.$$

Now, we would like to generalise $(*)$ even further to be able to deal with any partition Π. First remark that for blocks π and π' from Π_I, the above expressions for $\pi \trianglelefteq_I \pi'$ can be rewritten independently of the set I. For example, because $\mathcal{X}_I$ can be obtained from $\mathcal{X}$ by projection, we have that $(**)$ is equivalent to

$$\pi \trianglelefteq_I \pi' \iff (\exists \mathbf{x} \in \pi)(\exists \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}) \,.$$

Because the right part does not depend on the set I anymore, this expression seems a good candidate for the generalisation: for any partition Π, we could define for all $\pi, \pi' \in \Pi$ that

$$\pi \trianglelefteq^1 \pi' \iff (\exists \mathbf{x} \in \pi)(\exists \mathbf{y} \in \pi')(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}) \,. \tag{6.3.1}$$

We could likewise define generalisations based on the other expressions for $\pi \trianglelefteq_I \pi'$. Of course, not all of these generalisations are equivalent to one another, so for each one of them a lot of questions arise: Is this definition semantically sound? What are the properties of this relation? Are these properties meaningful? Can we build a "principle of partial dominance" on it? Are there other possible generalisations? Remark the similarities with the problem stated in Section 4.4.2, where we searched for an ordering of the intervals of a chain. In the same spirit as in that section, we will not try out every possibility, but rather directly construct an ordering that reflects a semantical idea.

Partitions. Assume that we know the partial evaluations of a for some subset $C_I \subseteq C$, and the partial evaluations of b for the criteria in $C_J \subseteq C$. How will we compare them? Or even more generally, we might only know for an object a that for each $c \in C_I$ it holds that $c(a) \in V_c \subseteq \mathcal{X}_c$, where V_c is not restricted to be a singleton as previously, in other words, we only know that the partial evaluations of a belongs to some subset $A \subseteq \mathcal{X}$, and that those of b belong to some $B \subseteq \mathcal{X}$. How do we compare a and b?
Our goal in this section is to establish a comparison between two objects a and b based on whatever information is available about either of them, as long as there is no possibility for their evaluations to be equal, i.e. the sets A and B that define our information are disjoint.
We can realise a comparison between two objects on the basis of such partial information by understanding more profoundly the idea underlying the principle of partial dominance preservation. In essence, it tries to establish a global comparison between two objects based on partial information. Since an objective global comparison can only be done based on the dominance relation, for which all information is required, we have to express partial dominance in terms of dominance, which on its turn is expressed in terms of the product order $\leq_{\mathcal{X}}$ on $\mathcal{X}$.

Semantics. What are the semantics we expect for a weak partial dominance relation $\trianglerighteq_p$ (where the subscript "p" refers to "partial")? Expressing the meaning of a *weak preference relation* (see p. 88) is always more easily done by expressing the meaning of the strict preference relation, the indifference relation and the incomparability relation. In our case, the semantics underlying the strict preference part of the weak partial dominance relation can be described as

$$a \lhd_p b \quad \text{should mean} \quad \begin{cases} b \text{ could be better than } a & \text{AND} \\ a \text{ can never be better than } b & \text{AND} \\ a \text{ is not partially indifferent to } b \end{cases} \tag{6.3.2}$$

where "partially indifferent" refers to indifference from a partial dominance point of view, it does not refer explicitly to how a and b should be finally ranked (see lower).
The first two demands can easily be expressed in terms of dominance:

$$\text{"}b \text{ could be better than } a\text{"} \quad \text{becomes} \quad \text{"possibly } a \lhd b\text{"}\ ,$$

and

$$\text{"}a \text{ can never be better than } b\text{"} \quad \text{becomes} \quad \text{"impossibly } b \lhd a\text{"}\ .$$

The third demand is a bit more tricky because it asks for a semantical description of the partial indifference relation. However, indifference is not really linked with the idea of dominance: in the definition of the weak dominance relation, two objects are indifferent if and only if they are equal, which does not bring us one step closer to its underlying semantics.
Instead of focussing on indifference, it is more interesting to put everything we already know together:

- We are looking for some weak preference relation $\trianglerighteq_p$,
- a is partially indifferent to b if $a \trianglelefteq_p b$ and $b \trianglelefteq_p a$,
- $\lhd_p$ is asymmetrical and $a \lhd_p b$ if $a \trianglelefteq_p b$ and not $b \trianglelefteq_p a$,
- the semantics for $\lhd_p$ are asymmetrical and contain "possibly $a \lhd b$" and "not possibly $b \lhd a$".

In view of this all, we see that the semantics

$$a \trianglelefteq_p b \quad \text{means} \quad \text{possibly } a \trianglelefteq b\,, \tag{6.3.3}$$

implies the semantics (6.3.2). Indeed, it implies that

$$\text{not } b \trianglelefteq_p a \quad \text{means} \quad \begin{cases} \text{impossibly } b \trianglelefteq a & \text{AND} \\ a \neq b & \text{AND} \\ a \text{ is not partially indifferent to } b\,, \end{cases}$$

so, if we define $\lhd_p$ in terms of $\unlhd_p$, we stumble upon the semantics (6.3.2). Remark that this also induces a semantics for indifference:

$$a \text{ is indifferent to } b \quad \text{means} \quad \begin{cases} \text{possibly } a \unlhd b & \text{AND} \\ \text{possibly } b \unlhd a\,. \end{cases}$$

At first sight, there seem to be no arguments why this semantics would be improper in this context. So let us see if the proposed semantics are still meaningful in combination with (#).

The semantics w.r.t. supervised learning. The impact of the described semantics on (#) result in

$$\text{possibly } a \unlhd b \quad \text{implies} \quad a \preccurlyeq_S b\,.$$

This can clearly be criticised. Indeed, why would we not allow incomparability if there is not enough information to either confirm that $a \unlhd b$, or to be really sure that $a \preccurlyeq_S b$. To answer this question, we must be totally aware of the context of supervised learning we are working in. This means that one of our objectives is to provide a decision algorithm that delivers results that are consistent in the eyes of the user(s) of the algorithm. As discussed in Section 4.5.3, this requires the results to be monotone, i.e. it must be assured that whenever $a \unlhd b$, the algorithm states that $a \preccurlyeq_S b$. So, if we only know that "possibly $a \unlhd b$", this also includes the case where we actually have $a \unlhd b$, and therefore the algorithm's output should be $a \preccurlyeq_S b$.

6.3.3 The local product ordering

The semantical expression (6.3.3) leads us to the following definition of a local dominance relation on non-overlapping sets:

Definition 6.3.1

Let C be a set of true criteria and $\mathcal{X} = \prod_{c\in C} \mathcal{X}_c$. Let A and B be two subsets of $\mathcal{X}$, i.e. A and B correspond to sets of possible evaluations for objects from Ω, with $A \cap B = \emptyset$. We define the **local product ordering** $\preceq_\ell$ as

LOCAL PRODUCT ORDERING.

$$A \preceq_\ell B \iff (\exists \mathbf{x} \in A)(\exists \mathbf{y} \in B)(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y})$$

with the conventions that

$$A \prec_\ell B \iff (A \preceq_\ell B) \wedge \neg(B \preceq_\ell A)\,,$$

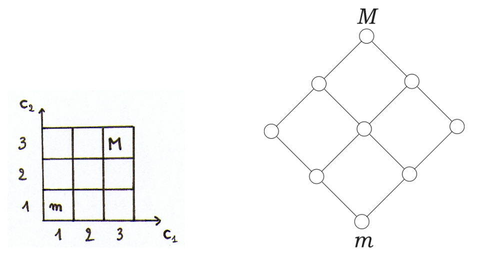
and

$$A -_\ell B \iff (A \preceq_\ell B) \wedge (B \preceq_\ell A)\,.$$

LOCAL DOMINANCE RELATION.

We can now define the **local dominance relation** in terms of the local product ordering. Consider two objects $a, b \in \Omega$ and assume that we only know that the evaluations of a belong to A, and the evaluations of b belong to B. We have $a \preceq_\ell b$ (resp. $a \prec_\ell b$ and $a -_\ell b$) if and only if $A \preceq_\ell B$ (resp. $A \prec_\ell B$ and $A -_\ell B$).

Why "local"? The term "local" originated from a graph theoretic point of view. The *poset* (see p. 52) $(\mathcal{X}, \leq_{\mathcal{X}})$ corresponds[1] to a lattice with a nice corresponding graph, see Figure 6.4.

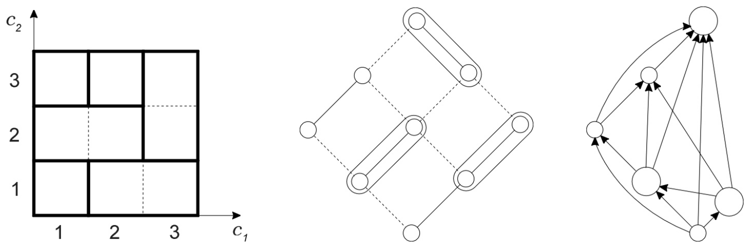


(a) The data space $\mathcal{X}$.

(b) The corresponding lattice.

Figure 6.4: $\mathcal{X}$ and the corresponding lattice.

If we consider a partition of $\mathcal{X}$, then the equivalence classes group together "indifferent" elements of $\mathcal{X}$. Now the problem is to define a kind of ordering on these blocks. If we look at the graph (see Figure 6.5), we see that several groups of nodes



(a) A partition of $\mathcal{X}$.

(b) Some of the nodes will be compressed.

(c) After compression.

Figure 6.5: The partitioning of $\mathcal{X}$ and the compression of the graph.

are replaced by a single node, these replacements are a kind of *local* compressions of the graph. As the product order was an order on the nodes of the initial graph, so is the local product ordering an ordering of these local compressions.

[1]If $\mathcal{X}$ is finite, otherwise, we need to consider the extended $\overline{\mathcal{X}} = \prod_{c \in C} \overline{\mathcal{X}}_c$ where $\overline{\mathcal{X}}_c = \mathcal{X}_c \cup \{\inf \mathcal{X}_c, \sup \mathcal{X}_c\}$.

Interpretation. The previous definition tells us only how to compare two objects. Essential to understanding this definition is that all objects within a subset $A \subseteq \mathcal{X}$ get the same treatment[2]. Consider for example Figure 6.6. Just following our

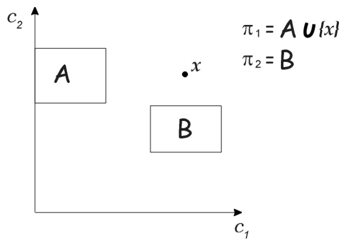


Figure 6.6: Interpretation of local product ordering

intuition one would be compelled to say that B outperforms A in a robust way: the evaluations in B score high on both criteria, while the evaluations in A only score high on one criterion, and there is only one exception, namely $\mathbf{x}$. However, if all evaluations in A must be treated the same and if $a \in A$ (i.e. the partial valuations of a belong to A) outperforms some object $b \in B$, then the objects of A should get a ranking at least as high as b. But since b gets the same ranking as all other objects in B, we must conclude that A should score better than B. The definition is such that there exists no possible evaluations to counter this reasoning.

Example 6.3.1. *Assume you want to build a dog kennel for your new hairy friend, the one with the huge puppy eyes that kept blinking at you at the pet store in Section 4.5.3 and for whose charmes you finally succumbed, as did your mellow hearted (girl/boy)friend/husband/wife when (s)he finally saw the puppy. But since you are also really fond of your furniture, you decided it would be best to keep this doggy out of the house as much as possible, so you really need a dog kennel. Of course, you want this kennel to be magnificent and you have to choose between some carpenters to help you construct it. There are two local carpenters, the good old Jimmy who is known to deliver decent work at a decent price, and some trendy newcomer P.J. of whom it is said the work is rather sloppy and the rates extreme. Then there is this very popular carpenter Rachem in your region who is known to be a a really good crafts(wo)man, and this at the same price as Jimmy. However, Rachem became so overbooked that she took an aid to help her out (whom you know to be not as good as trustworthy Jimmy), and you never know if you will get Rachem or her aid to help you out. Lastly, there is the carpenter with state renown, the best in the country, but with the associated price tag. So we have four groups A, B, C and D. Group A is good old Jimmy, B is trendy P.J., the group C is the*

[2]That is our basic assumption: if the current information only tells us that the evaluation of an object a belongs to A, then all evaluations belonging to A are *possible* evaluations for a. There is not one evaluation “more” possible than another one, i.e. we use the *characteristic function* χ_A (see p. 43) as a possibility distribution over the region A. This is in line with the idea of equivalence classes: inside an equivalence class, all elements are treated as one.

tandem Rachem and her aid, and finally D is the master carpenter. These groups are depicted in Figure 6.7.

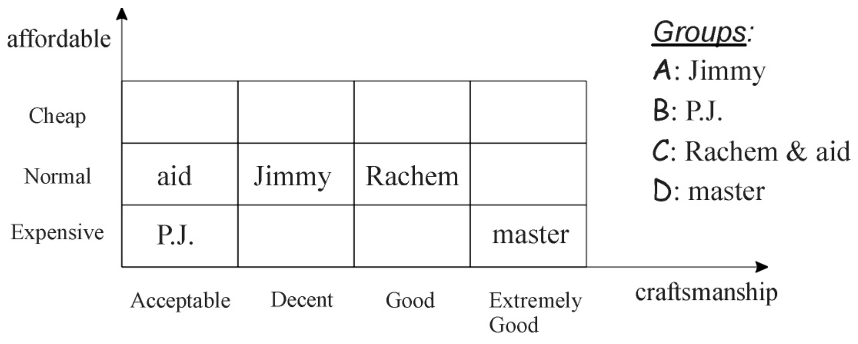


Figure 6.7: Building a doghouse.

We will now try to (completely) rank these groups. Clearly, B, trendy P.J., is the least interesting option. It is also obvious that there is no immediate preference between the groups A and D, and between C and D, their relative order is not fixed by what we know. How can we compare A, Jimmy, and C, Rachem and her aid? Rachem is has more qualities than Jimmy, so there are arguments against sustaining that C should be better than A. But Jimmy is preferred to Rachem's aid, providing an argument against putting A in front of C. Therefore, based on the current information, we can only plead indifference and give them the same rank. (Remark that with more information, A and C might be differentiated.) The two possible (complete) rankings are shown in Figure 6.8.

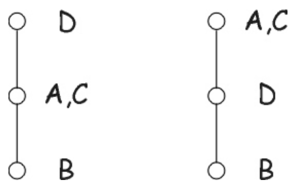
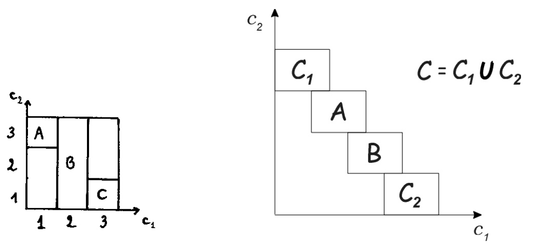


Figure 6.8: Ranking the carpenters (Hasse diagram).

About transitivity. Although the local product ordering is meaningful for the comparison of two blocks of a partition, it is not transitive. The following example will show this. Assume we are working with two criteria with values in $\mathcal{X}_{c_i} = \{1, 2, 3\}$, where $1 <_{c_i} 2 <_{c_i} 3$. Now consider the subsets of evaluations $A = \{(1, 3)\}$, $B = \{(2, 1), (2, 2), (2, 3)\}$ and $C = \{(3, 1)\}$. We have $A \prec_\ell B$ because $(1, 3) <_\mathcal{X} (2, 3)$ and $(1, 3)$ is incomparable to $(2, 1)$ and $(2, 2)$. Likewise we have $B \prec_\ell C$, but we do not have $A \prec_\ell C$ because $(1, 3) \parallel_\mathcal{X} (3, 1)$.



(a) Non-transitivity. (b) Cycles.

Figure 6.9: Non-transitivity and cycles in the local product ordering.

About cycles. Even though it will not influence our discussion in any sense, it is worth while to notice that the local product ordering may contain cycles. The simple example depicted in Figure 6.9(b) makes this clear: we have $C \prec_\ell A \prec_\ell B \prec_\ell C$. In Section 7.2.1 (see p. 186), we will come back to this in the context of decision trees.

About antisymmetry. Finally, it is worth while to emphasis that the local product ordering is not antisymmetric, i.e. $A -_\ell B$ does not correspond to $A = B$.

6.4 The principle of partial dominance preservation

6.4.1 Partial dominance

We would like to extend the principle of dominance preservation (6.2.2) by replacing the dominance relation with the local dominance relation. However, as we argued in Section 6.2.3, whatever relation replacing the dominance relation in (6.2.2) must be transitive to avoid problems with the axiom of independence of irrelevant alternatives. This condition is not met by the local dominance relation. We also argued that this problem can be overcome by considering the transitive closure. This leads us to the following definitions:

Definition 6.4.1

LOCAL PRODUCT PREORDER.

The **local product preorder** $\lesssim_\ell$ on non-intersecting subsets of $\mathcal{X}$ is the transitive closure of the local product ordering $\preceq_\ell$. (We follow the usual conventions in writing $<_\ell$ and $\sim_\ell$.)

PARTIAL DOMINANCE RELATION.

The **partial dominance relation** $\trianglelefteq_p$ is now defined in function of the local product order. Let $a, b \in \Omega$ and assume we only know that the partial evaluations of a belong to $A \subseteq \mathcal{X}$, and those of b belong to $B \subseteq \mathcal{X}$, with $A \cap B = \emptyset$, then we have that $a \trianglelefteq_p b$ if and only if $A \lesssim_\ell B$. The subscript "p" refers to the "partial" knowledge we have about a and b.

Obviously, the local product preorder is indeed an preorder: it is reflexive and transitive. The fact that it is in general not antisymmetric will have some repercussions as we will see in Section 6.5.3.

6.4.2 The principle

PRINCIPLE OF PARTIAL DOMINANCE PRESERVATION.

The **principle of partial dominance preservation** is a straightforward extension of the principle (6.2.1):

$$a \lhd_p b \implies \lambda(a) \leq_{\mathcal{L}} \lambda(b).$$

This principle can be interpreted as follows: "*if for two objects $a, b \in \Omega$, we know that based on the information we have (or take into account) about a and b, a might be dominated by b, but not vice-versa, then the ranking of b must be at least as high as the ranking of a*".

Question. Recall from Section 6.2.3 that the principle of dominance preservation is based on the weakened form $a \lhd b \Rightarrow a \preccurlyeq_S b$ of the basic MCDA principle $a \unlhd b \Rightarrow a \preccurlyeq_S b$. We know this weakened form is meaningful in combination with the dominance relation, but is it still meaningful in combination with other preference relations, like for example the partial dominance relation?

Doubt and reversed preference Already in Chapter 2, we mentioned the notion of *doubt* (see p. 32), two objects $a, b \in \Omega$ with $\mathbf{a} = \mathbf{b}$ but $\lambda(a) \neq \lambda(b)$. Then in Chapter 4, the notion of *reversed preference* (see p. 100) was introduced. Reversed preference arises between the ranking $(\lambda, \leq_{\mathcal{L}})$ and the set of criteria when the principle of dominance preservation is violated.
Although doubt and reversed preference were introduced separately from each other, and although they treat very different problematics[3], they do in fact, in the context of rankings, arise from the same basis, namely violation of the basic principle

$$a \unlhd b \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)\,. \tag{##}$$

Indeed, (##) can be decomposed into

$$\begin{cases} (a \unlhd b) \wedge (b \unlhd a) & \Rightarrow \lambda(a) = \lambda(b) \\ \qquad a \lhd b & \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)\,, \end{cases}$$

and violation of the first part results in doubt (because we have $(a \unlhd b) \wedge (b \unlhd a)$ if and only if $\mathbf{a} = \mathbf{b}$), while violation of the second part results in reversed preference.
Based on the previous, it is child's play to generalise the notions of doubt and reversed preference between the ranking $(\lambda, \leq_{\mathcal{L}})$ and the set of criteria C towards the use of any integrated (weak) dominance relation, and hence in particular for $\unlhd_p$, the (weak) partial dominance relation.

[3]Remember that doubt can be tolerated in a ranking, while reversed preference is never acceptable (see p. 100).

Definition 6.4.2

Let $\preccurlyeq_D$ be an integrated dominance relation.

DOUBT.

(i) There is **doubt** between the ranking $(\lambda, \leq_{\mathcal{L}})$ and $\preccurlyeq_D$ if

$$(\exists(a,b) \in \Omega^2)(a \sim_D b \wedge \lambda(a) \neq \lambda(b))\,.$$

REVERSED PREFERENCE.

(ii) There is **reversed preference** between the ranking $(\lambda, \leq_{\mathcal{L}})$ and $\preccurlyeq_D$ if

$$(\exists(a,b) \in \Omega^2)(a \prec_D b \wedge \lambda(a) \not\leq_{\mathcal{L}} \lambda(b))\,.$$

Remark that what we called "doubt, resp. reversed preference, between the ranking and the set of (true) criteria" in Definition 4.5.1 (see p. 97), resp. Definition 4.5.2 (see p. 100), corresponds to "doubt, resp. reversed preference, between the ranking and the dominance relation".

6.4.3 An example

Example (part 3). We already explained why the first split should be based on c_2. With the definition of the partial dominance relation, it is now also possible to show graphically that the split based on c_2 is better than a split based on c_1 (or c_3), as can be seen in Figure 6.10. The possibilities for the second split are depicted in

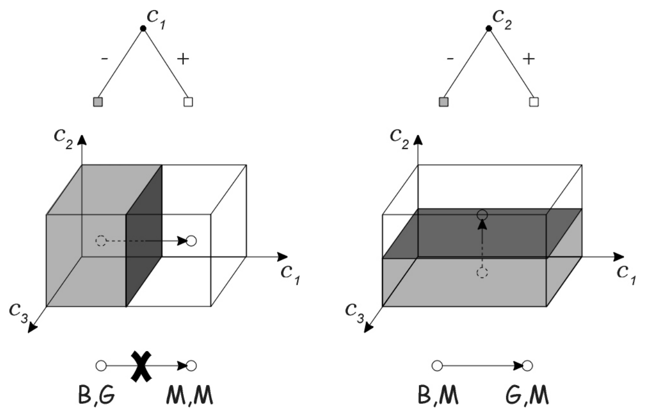


Figure 6.10: The splits based on c_1, c_2, along with the induced partition of $\mathcal{X}$ and the corresponding (locally compressed) graph. The arrows in the graph show the local product ordering $<_\ell$. We cross out the arrow if the principle of partial dominance preservation is violated.

Figure 6.11. We clearly see that the split based on c_3 leads to a violation of the principle of partial dominance preservation: $a_2 \lhd_p a_1$ and $\lambda(a_2) = \mathsf{M} \not\leq_{\mathcal{L}} \lambda(a_1) = \mathsf{B}$. The split based on criterion c_2 is in line with the principle and is therefore chosen.

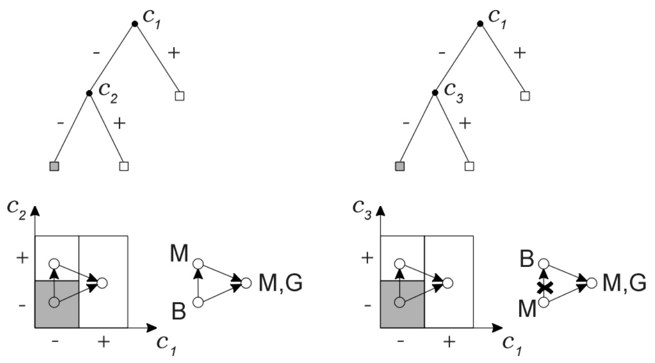


Figure 6.11: Two possibilities for the second split: trees, induced partitions and graphs.

6.5 Monotone classifiers

6.5.1 Introduction

Up to this moment, the starting point in the previous sections was the initial ranking $(\lambda, \leq_{\mathcal{L}})$ of the objects from Ω and not some representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ of a ranking as discussed in Chapter 4. A reasonable question is why did we bother to do this if, as discussed in Chapter 4, *"for practical purposes we use the representation of the ranking"*, and *"we must only take care of reversed preference inside the representation"* (see p. 102).
The answer is simple: sometimes we want to represent our ranking by a model, and while representation is an important aspect of modelling, it is only *one* aspect of it. In the end, a model will be a representation of whatever we tried to model, so *comparing* models amounts to comparing representations. However, while *building* these models, we must take into account certain aspects of the data that will be smoothed in the final model representation. For example, we can compare the rule bases derived from a tree based approach and a rough set based approach (see Appendix 1.4), but they are both built in a different way.

6.5.2 Representations based on partitions

In Chapter 2 we emphasised the importance of partitions for classifiers with (the very simple) Proposition 2.A.1 (see p. 48). Now denote by $\lambda_{\text{cl}} : \mathcal{X} \to \lambda_{\text{cl}}(\mathcal{X})$ a (static, i.e. time-independent) classifier without random component that works on Ω via its representation $\mathcal{X}$. Clearly λ_{cl} induces a partition Π_{cl} on $\mathcal{X}$ such that all elements inside the same block are mapped to the same output. This means we can define $\lambda_{\text{cl}} : \Pi_{\text{cl}} \to \lambda_{\text{cl}}(\mathcal{X})$ by

$$\lambda_{\text{cl}}(\pi) := \lambda_{\text{cl}}(\mathbf{x}),$$

where $\mathbf{x} \in \pi \in \Pi_{\text{cl}}$.

In the remainder of this section, we will write λ_{repr} instead of λ_{cl}, because it only refers to the representational part of the classifier, and not to the classifier itself, which gives us a more general setting: even though any possible representation of a classification can be seen as a classifier, it is clear that the known classifiers do not comprise all possible representations.

6.5.3 Partition-based monotonicity

Let $(\lambda_{\text{repr}}, \trianglelefteq)$ be any *representation of a ranking* (see p. 92) $(\lambda, \leq_{\mathcal{L}})$. In Section 4.5, we defined the *elementary monotonicity constraint* (see p. 100):

$$\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})\,.$$

We are now able to define another version of this constraint taking into account that all representations are somehow partition-based. Let Π_{repr} denote the partition induced by λ_{repr}, then we obtain the following **partition-based monotonicity constraint**:

PARTITION-BASED MONOTONICITY CONSTRAINT.

$$\boxed{\pi_1 \lesssim_\ell \pi_2 \Rightarrow \lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2)}$$

CONSISTENT W.R.T. THE INDUCED PARTITION.

A representation of a ranking is said to be **consistent w.r.t. the induced partition** if it satisfies the partition-based monotonicity constraint.
Remark that it demands far less effort to validate the partition-based monotonicity constraint than the elementary one because only the blocks of the partitions need to be checked, and not all vectors in $\mathcal{X}$. This can be a huge asset in practice. Moreover, the following lemma shows that if partition-based monotonicity is satisfied, we immediately have elementary monotonicity, which is of course no real surprise considering our discussion on the impact of the semantics underlying $\trianglelefteq_p$ on (##). But there is more, Lemma 6.5.2 shows that in any ordinary situation, the partition-based monotonicity constraint can be written in a weaker form, and Proposition 6.5.3 shows that – again in any normal situation – these two forms of monotonicity actually coincide!

Lemma 6.5.1. *The partition-based monotonicity constraint implies the elementary monotonicity constraint. In other words, a representation of a ranking that is consistent w.r.t. its induced partition is a consistent (see p. 100) representation.*

Proof.
Suppose that we have $\pi_1 \lesssim_\ell \pi_2 \Rightarrow \lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2)$. Assume that $\mathbf{x} <_{\mathcal{X}} \mathbf{y}$, and denote $\Pi(\mathbf{x}) = \pi_{\mathbf{x}}$ and $\Pi(\mathbf{y}) = \pi_{\mathbf{y}}$. By definition, we have $\pi_{\mathbf{x}} \preceq_\ell \pi_{\mathbf{y}}$, implying $\lambda_{\text{repr}}(\mathbf{x}) = \lambda_{\text{repr}}(\pi_{\mathbf{x}}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_{\mathbf{y}}) = \lambda_{\text{repr}}(\mathbf{y})$. □

Lemma 6.5.2 ([4]). *In case $\trianglelefteq_{Im}$ is transitive, we can replace the monotonicity condition by the following equivalent definition:*

$$\pi_1 \preceq_\ell \pi_2 \Rightarrow \lambda_{repr}(\pi_1) \trianglelefteq_{Im} \lambda_{repr}(\pi_2)\,.$$

Proof.
Assume $\pi_1 \lesssim_\ell \pi_2$. If $\pi_1 \not\preceq_\ell \pi_2$, then there exists a sequence $(\pi'_i)_{i=1}^k$ in Π_{repr} such that

$$\pi_1 = \pi'_1 \preceq_\ell \ldots \preceq_\ell \pi'_k = \pi_2 \,.$$

This means that

$$\lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \ldots \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2)\,,$$

and transitivity now leads to $\lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2)$. □

We now show that under two very natural additional conditions, namely that the ordering $\trianglelefteq_{\text{Im}}$ on $\lambda_{\text{repr}}(\mathcal{X})$ is also[5] transitive and that $\triangleleft_{\text{Im}}$ contains no cycles, these two monotonicity constraints are equivalent to each other:

Proposition 6.5.3. *Let $(\lambda_{repr}, \trianglelefteq_{Im})$ be a representation of the ranking $(\lambda, \leq_{\mathcal{L}})$, and let Π_{repr} be the partition induced by λ_{repr}. If $\trianglelefteq_{Im}$ is transitive and $\triangleleft_{Im}$ contains no cycles, then the partition-based monotonicity constraint and the elementary monotonicity constraint are equivalent, i.e.*

$$\begin{gathered}(\forall \mathbf{x}, \mathbf{y} \in \mathcal{X})(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{repr}(\mathbf{x}) \trianglelefteq_{Im} \lambda_{repr}(\mathbf{y}))\\ \Updownarrow \\ (\forall \pi_1, \pi_2 \in \Pi_{repr})(\pi_1 \lesssim_\ell \pi_2 \Rightarrow \lambda_{repr}(\pi_1) \trianglelefteq_{Im} \lambda_{repr}(\pi_2))\,.\end{gathered}$$

Proof.
(i) That partition-based monotonicity implies elementary monotonicity was already proven in Lemma 6.5.1.
(ii) Now assume that $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})$ holds. Because of the preceding Lemma 6.5.2, we only need to investigate the case when $\pi_1 \preceq_\ell \pi_2$.

(a) $\pi_1 \prec_\ell \pi_2$. This means there exist an $\mathbf{x} \in \pi_1$ and a $\mathbf{y} \in \pi_2$ such that $\mathbf{x} <_{\mathcal{X}} \mathbf{y}$. By assumption, this leads to $\lambda_{\text{repr}}(\pi_1) = \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y}) = \lambda_{\text{repr}}(\pi_2)$.

(b) $\pi_1 -_\ell \pi_2$. This means that $\pi_1 \preceq_\ell \pi_2 \preceq_\ell \pi_1$. So there exist $\mathbf{x}, \mathbf{x}' \in \pi_1$ and $\mathbf{y}, \mathbf{y}' \in \pi_2$ such that $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ and $\mathbf{y}' \leq_{\mathcal{X}} \mathbf{x}'$. By assumption, this leads to $\lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y}) = \lambda_{\text{repr}}(\mathbf{y}') \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{x}') = \lambda_{\text{repr}}(\mathbf{x})$, or, because $\triangleleft_{\text{Im}}$ does not contain cycles, $\lambda_{\text{repr}}(\mathbf{x}) \sim_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})$, from which $\lambda_{\text{repr}}(\pi_1) \sim_{\text{Im}} \lambda_{\text{repr}}(\pi_2)$. □

As a consequence, we have that the previous holds for all orderings $\trianglelefteq_{\text{Im}}$ of the image of λ_{repr} we considered in Chapter 4:

Corollary 6.5.4. *For any representation $(\lambda_{repr}, \trianglelefteq_{Im})$ where $\trianglelefteq_{Im}$ is either the total order $\leq_{\mathcal{L}}$ on $\mathcal{L}$, the partial order $\leq^{[2]}$ on $\mathcal{L}^{[2]}$ (see p. 96) or the weak first order stochastic dominance $\trianglelefteq_{(1)}$ (see p. 106), we have that partition-based monotonicity is the same as elementary monotonicity.*

[5]Besides the conditions demanded in Section 4.4.2, namely reflexivity, proper semantics, independence of λ and that it should extend $\leq_{\mathcal{L}}$. Remark however that we do not need any of these basic properties for the proof of Proposition 6.5.3.

A remark concerning antisymmetry. Note that the elementary monotonicity constraint has two equivalent forms:

$$\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})$$

and

$$\mathbf{x} <_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})\,,$$

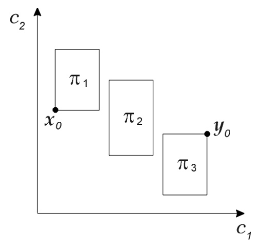
but we do not have such a property for the partition-based monotonicity constraint. Moreover, we have

$$(\forall \mathbf{x}, \mathbf{y} \in \mathcal{X})(\mathbf{x} <_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})) \tag{6.5.1}$$

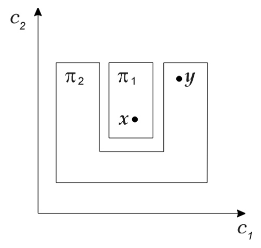
$$\not\Downarrow \quad \not\Uparrow$$

$$(\forall \pi_1, \pi_2 \in \Pi_{\text{repr}})(\pi_1 <_{\ell} \pi_2 \Rightarrow \lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2))\,, \tag{6.5.2}$$

not even when $\trianglelefteq_{\text{Im}}$ is transitive and $\triangleleft_{\text{Im}}$ contains no cycles. This can be easily seen in Figure 6.12. Firstly, figure (a) contradicts the downward implications: if $(\forall a \in \Omega)(\mathbf{a} \in \pi_1 \Rightarrow \lambda(a) = 2)$ and $(\forall a \in \Omega)(\mathbf{a} \in \pi_3 \Rightarrow \lambda(a) = 1)$, and if $(\forall a \in \Omega)(\mathbf{a} \in \pi_2 \Rightarrow ((\mathbf{a} >_{\mathcal{X}} \mathbf{x}_0 \Rightarrow \lambda(a) = 2) \wedge (\mathbf{a} <_{\mathcal{X}} \mathbf{y}_0 \Rightarrow \lambda(a) = 1)))$ then the elementary monotonicity constraint is satisfied, while $\pi_1 <_{\ell} \pi_3$ and[6] $\hat{\lambda}_{\Pi}(\pi_1) = 2 >_{\mathcal{L}} \hat{\lambda}_{\Pi}(\pi_3) = 1$. Figure (b) contradicts the upward implication: because $\pi_1 \sim_{\ell} \pi_2$, we cannot say anything about $\lambda_{\text{repr}}(\mathbf{x})$ and $\lambda_{\text{repr}}(\mathbf{y})$.



(a) Counterexample.



(b) Counterexample.

Figure 6.12: Counterexamples.

It should be noted that the culprit behind the second counterexample is that $\lesssim_{\ell}$ is not necessarily antisymmetric (i.e. partial indifference $\sim_{\ell}$ may be different from equality $=$). Indeed, the following self-evident lemma holds:

Lemma 6.5.5. *If the partition Π_{repr} is such that $\lesssim_{\ell}$ behaves as an antisymmetric relation on its blocks, we have that (6.5.2) is equivalent to the partition-based monotonicity constraint.*

Corollary 6.5.6. *Under the conditions of Lemma 6.5.5, we have that (6.5.1) implies (6.5.2). Under the additional conditions of Proposition 6.5.3, it holds that expressions (6.5.1) and (6.5.2) are equivalent.*

[6]Here $\hat{\lambda}_{\Pi}$ is defined as in Section 2.3.3, Equation (2.3.1) (see p. 35).

Doubt and reversed preference. In case we can not rely on the property of antisymmetry, there can exist doubt in the representations $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ of a ranking $(\lambda, \leq_{\mathcal{L}})$ that is not present in the representation λ_{repr} of the classification λ. So we end up with the following definitions that have to be distinguished from the ones in Definition 6.4.2:

Definition 6.5.1

(i) There is **doubt** inside the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ if

DOUBT.

$$(\exists(\pi_1, \pi_2) \in \Pi_{\text{repr}})(\pi_1 \sim_\ell \pi_2 \wedge \lambda_{\text{repr}}(\pi_1) \not\sim_{\text{Im}} \lambda_{\text{repr}}(\pi_2))\,.$$

(ii) There is **reversed preference** inside the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ if

REVERSED PREFERENCE.

$$(\exists(\pi_1, \pi_2) \in \Pi_{\text{repr}})(\pi_1 <_\ell \pi_2 \wedge \lambda_{\text{repr}}(\pi_1) \ntrianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2))\,.$$

6.5.4 Consistency

We wind this section up with reconsidering the main results of Chapter 4. We start by relaxing the condition of *minimal consistency* (see p. 107)

Definition 6.5.2

We say that the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ of $(\lambda, \leq_{\mathcal{L}})$ is **minimally consistent w.r.t. the induced partition** Π_{repr}, if for all $\pi_1, \pi_2 \in \Pi_{\text{repr}}$ with $\pi_1 \lesssim_\ell \pi_2$, it holds that

MINIMALLY CONSISTENT W.R.T. THE INDUCED PARTITION.

$$\hat{\lambda}_r(\pi_1) \leq_{\mathcal{L}} \hat{\lambda}_\ell(\pi_2) \implies \lambda_{\text{repr}}(\pi_1) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\pi_2)\,,$$

where

$$\begin{aligned} \hat{\lambda}_\ell(\pi) &= \min\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} \in \pi\}\,, \\ \hat{\lambda}_r(\pi) &= \max\{\lambda(a) \mid a \in \Omega \wedge \mathbf{a} \in \pi\}\,. \end{aligned}$$

Lemma 6.5.7. *If $(\lambda_{repr}, \trianglelefteq)$ is minimally consistent w.r.t. its induced partition, then it is also minimally consistent in $\mathcal{X}$.*

Proof.
Assume $\hat{\lambda}_r(\pi_1) \leq_{\mathcal{L}} \hat{\lambda}_\ell(\pi_2)$. Because $(\forall \mathbf{x} \in \pi_1)(\hat{\lambda}_r(\mathbf{x}) \leq_{\mathcal{L}} \hat{\lambda}_r(\pi_1))$ and $(\forall \mathbf{y} \in \pi_2)(\hat{\lambda}_\ell(\pi_2) \leq_{\mathcal{L}} \hat{\lambda}_\ell(\mathbf{y}))$, we find $\hat{\lambda}_r(\mathbf{x}) \leq_{\mathcal{L}} \hat{\lambda}_\ell(\mathbf{y})$ leading to $\lambda_{\text{repr}}(\pi_1) = \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq \lambda_{\text{repr}}(\mathbf{y}) = \lambda_{\text{repr}}(\pi_2)$. □

This enables us to weaken Proposition 4.7.2 to the following statement:

Corollary 6.5.8. *Substituting the ranking $(\lambda, \leq_{\mathcal{L}})$ by a representation $(\lambda_{repr}, \trianglelefteq_{Im})$ that is minimally consistent w.r.t. its induced partition will never introduce new reversed preferences.*

Consistent representations. And now we come to the two theorems of Chapter 4, Theorem 4.6.3 (see p. 104) about the consistent interval representation, and Theorem 4.7.3 (see p. 108) about a consistent stochastic representation. Both theorems and their proofs can be immediately generalised towards the context of partitions. Indeed, their proofs rely on the fact that for $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$, it holds that $(\mathbf{x}] = \{\mathbf{x}' \in \mathcal{X} \mid \mathbf{x} \leq_{\mathcal{X}} \mathbf{x}'\} \subseteq (\mathbf{y}]$ and $[\mathbf{y}) \subseteq [\mathbf{x})$. Because of the transitivity of $\lesssim_{\ell}$, we have this same property for $\pi_1, \pi_2 \in \Pi$ with $\pi_1 \lesssim_{\ell} \pi_2$.
Lemma 6.5.1 now assures the resulting partition-based representations are still monotone in the usual sense! In particular, this means that the OSDL algorithm (Ordinal Stochastic Dominance Learner, Chapter 5), can be applied on any partition of the data space $\mathcal{X}$ using the local product preorder instead of the product order.

6.6 Future research

One of our basic assumptions is that we only consider disjoint sets for defining the local product ordering. An interesting extension would be to investigate the impact of allowing intersecting sets. Still in the same direction, it would be worthwhile to extend the setting even further towards graded relations.

Interlude

RANKING TREES, THE MAKING OF
A STORY BEHIND THE SCENES

YOUNG HEARTS. *Once upon a time, in the year 2001, while the short dark cold winter months gradually were easing into the softer tones and spirits of a promising spring, an ardent young man deemed his research was ripe enough to be released out of its confinement and be cast into the vast and vibrant world. With the rash confidence that is proper to the verdancy of youngsters, he wrote and submitted proudly an extended abstract (12 pages) about ranking trees, based upon insight gained from his newly developed framework. Now he only needed to wait for a notice of acceptance.*

OHOH, PROBLEMS... *Time whirled the consecutive months nimbly off, until I finally received an email from the conference organisers. I was not just a bit disillusioned when I discovered I had been refused. But an even greater shock was about to blast me away: in the session referee's comments attached to the email, it was rather bluntly and ungainly hinted that I had no knowledge of the related literature whatsoever and that everything I presented in my paper had already been done and published previously by others. Aaaaarrrrggghhhhh!!!!!! A whole year of hard work and intensive labour down the drain, all in vain, all for nothing. A quick look up on the internet of the adjoined references they so kindly and mercifully provided, endorsed the referee's comments and gave me plenty of reasons to get in an even bleaker mood.*

I think that was the swiftest plunge into a Ph.D. blues I ever took. I still remember the email I sent to Rob Potharst, one of the authors of the reference articles I was given, to beseech him to forward me his Ph.D. thesis on the topic of monotone classification. I also remember vividly his warm and encouraging response to my distressed (and probably depressed) words. Bernard reacted more stoical to this situation. But then again, he always kept the utmost confidence in me and my progressions, even when I was totally stripped of it myself. It seems, luckily, that he was right and I was wrong.

THE RECOVERY. *Indeed, when I started to study these reference articles more closely, it began to dawn on me that although there was some resemblance in a few of the ideas, my approach was fundamentally different. The referee had obviously made the mistake of jumping a bit too diagonal over the paper. Relief washed through me. In the end, I salvaged the paper somewhat by rewriting it in light of this new information and sending it to a journal where it finally got accepted.*

HOBBLING ALONG. *All went relatively well from that moment on. I cheerfully submitted another handful of papers to machine learning and data mining conferences (on various topics ranging from my classification framework, over the OSDL algorithm, to impurity measures for ranking), and yes, I got pleasantly refused for all of them. Isn't that nice? Aah, and also Chapter 4 was refused as a contribution for some journal's special issue. Oh well, after a while, you just get used to those congenial emails with* "We are sorry to inform you that...", *and from the reports I usually had to conclude that they didn't really grasp my ideas, which is also useful information (whether it's them or my clumsy writing style, I will leave aside, probably a bit of both). I learned the hard way what people – or at least the referees I encountered – seem to expect. Stated baldly, they like everything that is written in the form of theorems and proofs, they prefer dry text, they prefer ϵ-contributions*[7] *subdued to extensive experimental testing and they do not like (or do not understand?) discussion about semantical foundations, nor theoretical and abstract papers without any application (unless in theorem+proof style). Oh, and not unimportant, I guess many of the referees were not really fond of the topic (it's not a very popular one indeed, although that appears to be changing ever so slightly over the years). Anyway, I took some of these hints to heart, the ones from which I could see the surplus value, like the theorem+proof style that, when practiced judiciously, relentlessly captures the attention of the unheeding reader, and I stubbornly disregarded the other ones (maybe one day I will see the wisdom in them, and laugh heartily at my headstrong demeanour and unrelenting stupidity... but so far, I don't).*

OHOH, AGAIN... *Not so long ago, at the beginning of April this year, when a merry and pleasant spring had just taken over the reign of a forbidding but bearable winter, I estimated I but had to jot all my findings down on paper and be done with it. But didn't I already mention that I am quite a lousy estimator? Well, all went exactly according to the plan I had sketched, giving me plenty of time with a broad three months for writing until the end of June, and another month of editing and drawing figures in August. Until that dreadful moment when I discovered that my final ranking tree algorithm did not do in all circumstances what it was supposed to do. Aaaaarrrrggghhhhh!!!!!! In the mean time, I already had quite some expertise in dealing with the Ph.D. blues, but it still was a very unpleasant finding, nagging me and depriving me of full sleep. Bernard reminded me that the algorithm did not produce wrong output, only that it was a bit less performing than what could have been expected. And he assured me that this did not jeopardise my dissertation in any way. But for me, this algorithm was the finishing touch of my Ph.D., it had been the driving force during all these years and I would consider my thesis as incomplete and unsatisfactory if it would not be precisely working as I originally intended it.*

[7] A small contribution on existing and well-known topics – *Bernard De Baets' Unabridged Dictionary of Personal Scientific Language*.

THE REPARATION. *It didn't take that long to find the missing stitch and it wasn't even difficult to sew it back together. However, like in clothing, it is all about the details:* "Hey, that seam is on the outside. Shouldn't it have been on the inside?" *And since my own research philosophy prohibits me from doing something I do not understand to its full depth, I had to unstitch everything, and even unravel the pieces of cloth themselves to get to the bottom of it... It took me one very stressful month and a half... just to understand it, not even to write it down. And it is common knowledge that entrusting new insights to paper tends to be a lot slower than scribbling away well digested information. All in all, my agreeable three months for writing had crumpled down to a pathetic month and a half. That's a story that I already told throughout the other interludes. And here is another one of these diary entries, written down during the struggle with Chapter 7.*

SATURDAY, 31 MAY, 2003. *These days, I am not just preparing my Ph.D., I am also preparing for the 5-ball cascade, that's juggling with 5 beautiful and colourful balls. (Since half of April, I suddenly got this renewed interest in juggling that grew even more intense since I began writing 2 weeks ago). Just half an hour ago, I laid down my continuous chapter writing, to take up my continuous ball juggling. (That'ssss a 3-ball trick in juggling alsssso called "the ssssnake", becausssse the ballssss form a continuousssss pattern, assss in the popular game with the sssame name. It's really pretty to look at, and a perfect exercise for the 5-ball cascade because it's the same throws but with two balls missing). Anyway, yesterday I read that if you get stressed with all these balls in the air, you should* "try to visualise the figure like a unit rather "living" than like a succession of balls [Arlabosse]"*. And today I discovered – to my own wonder I may add – that this works wonderfully well. It's a nice example of how a different perspective can ease things up. Let's see if I can adapt this new insight into the writing of my thesis ;-) believe it or not, but that was in fact one of the first things that popped up in my mind while keeping control of my 3-ball bouncing beast (how far can someone be gone?)*

References

[Arlabosse] *D. Arlabosse,* Juggling workshop: 5 balls*, [Online]. Available: http://didier.arlabosse.free.fr/balles/english/index.html.*

Chapter 7

The basics of ranking trees

Notions and conventions.

CLASSIFICATION. CLASSIFICATION. A **classification** (see p. 29) is a mapping $\lambda : \Omega \to \mathcal{L}$, assigning class labels to a set of objects. Each object $a \in \Omega$ is described by a finite set of *attributes* $q : \Omega \to \mathcal{X}_q$ (see p. 30), and can therefore be represented by a vector $\mathbf{a} = (q_1(a), \ldots, q_n(a)) \in \mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_{q_i}$. This induces a *representation* λ_{repr} of the classification λ (see p. 31), mapping vectors from $\mathcal{X}$ to some set of labels. We speak of **doubt** (see p. 32) if $\mathbf{a} = \mathbf{b}$ (implying $\lambda_{\text{repr}}(\mathbf{a}) = \lambda_{\text{repr}}(\mathbf{b})$) and $\lambda(a) \neq \lambda(b)$. DOUBT.

WEAK PREFERENCE RELATION $\preccurlyeq$. PREFERENCE MODELLING. A **weak preference relation** [118] (see p. 88) is a reflexive relation $\preccurlyeq$ where the expression $a \preccurlyeq b$ stands for "a is at most as good as b". A function $f : (X, \preccurlyeq_X) \to (Y, \preccurlyeq_Y)$ between two sets equipped with a preference relation is called **monotone** (see p. 52) if for all $x, y \in X$ it holds that MONOTONE FUNCTION.

$$x \preccurlyeq_X y \Rightarrow f(x) \preccurlyeq_Y f(y)\,.$$

ORDER. An **order** (see p. 52) $\leq_X$ on a set X is a binary relation that is *reflexive* ($x \leq_X x$), *antisymmetric* (if $x \leq_X y$ and $y \leq_X x$, then $x = y$) and *transitive* (if $x \leq_X y$ and $y \leq_X z$, then $x \leq_X z$). It is in particular a weak preference relation. The couple $(X, \leq_X)$ is called a **poset** (partially ordered set). The subset $[x, y] = \{z \in X \mid x \leq_X z \leq_X y\}$ is called a **(poset) interval** in $(X, \leq_X)$. An order is called **complete** if for all $x, y \in X$ either $x \leq_X y$ or $y \leq_X x$. POSET $(X, \leq_X)$. (POSET) INTERVAL $[x, y]$. COMPLETE.

CRITERION. MULTI-CRITERIA DECISION AID. A **criterion** [97] (see p. 98) is defined as a mapping $c : \Omega \to (\mathcal{X}_c, \leq_c)$, where $\leq_c$ is a complete order on $\mathcal{X}_c$, such that it appears meaningful to compare two objects a and b, according to a particular point of view, on the sole basis of their evaluations $c(a)$ and $c(b)$. We will only consider so-called *true criteria* [20]: a is preferred to b according to criterion c if $c(a) >_c c(b)$. Moreover, we assume that all $\mathcal{X}_c$ are finite and that the underlying scale of all criteria is ordinal.

COMPLETE RANKING $(\lambda, \leq_{\mathcal{L}})$. RANKING. A **complete ranking** $(\lambda, \leq_{\mathcal{L}})$ (see p. 90) consist of a classification λ and a complete order $\leq_{\mathcal{L}}$ on $\mathcal{L}$ that induces a preference relation on Ω: $a \preccurlyeq b$ if $\lambda(a) \leq_{\mathcal{L}} \lambda(b)$. When the objects are described by a finite set of criteria, $C = \{c_i \mid i \in N = \{1, \ldots, n\}\}$, then this induces a *representation* $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ (see p. 92). The product order $\leq_{\mathcal{X}}$ on $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_{c_i}$ is defined by $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$ if $(\forall i \in N)(x_i \leq_{c_i} y_i)$.

7.1 Partition-based modelling of a ranking

Notions and conventions.

PARTITION. PARTITIONS. A **partition** Π of $\mathcal{X}$ is a set of non-empty, pairwise disjoint subsets π of $\mathcal{X}$ such that $\bigcup_{\pi \in \Pi} \pi = \mathcal{X}$. The elements π of a partition Π are called **blocks**. The unique block containing $\mathbf{x} \in \mathcal{X}$ is denoted by $\Pi(\mathbf{x})$. We also denote $\pi_\Omega = \{a \in \Omega \mid \mathbf{a} \in \pi\}$ and $\Pi_\Omega = \{\pi_\Omega \mid \pi \in \Pi\}$. BLOCKS. π_Ω, Π_Ω

LEARNING SAMPLE $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$. DATA SET. MODELLING. MODELLING. For rankings, the basic supervised learning problem is to induce a ranking $(\lambda, \leq_{\mathcal{L}})$ (or some representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ of it) from a given **learning sample** (also called **data set**) $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$, where $\mathcal{S} \subseteq \Omega$ and $d = \lambda_{|\mathcal{S}}$. This induction process is called **modelling**, or **building a model**.

PARTITION-BASED MODELLING, what does it mean? Firstly, mark the difference between the verb "modelling" and the result of this action, a "model". As we mentioned already in Appendix 2.A (see p. 48), any model can be interpreted as being partition-based (indeed, any representation λ_{repr} induces a partition on $\mathcal{X}$ and Ω, grouping all elements mapped to the same output). However, the process of building this model may not be partition-based in itself. We mean that once the data space $\mathcal{X}$ has been fixed (i.e. once the set of attributes/criteria has been fixed), no other[1] partitions were constructed during the building of the model. For example, building the k-Nearest Neighbour model is just putting the learning sample to memory, fixing the metric and identifying the number of neighbours to take into consideration. On the other hand, building a tree model is a clear example of partition-based modelling.

7.1.1 Ordering partitions

Recapitulation Chapter 6

The **local product preorder** $\lesssim_\ell$ (see p. 166) on the blocks of a partition Π of $\mathcal{X}$ is the transitive closure of the **local product ordering** $\preceq_\ell$ (see p. 162) which is defined by

LOCAL PRODUCT PREORDER $\lesssim_\ell$.

LOCAL PRODUCT ORDERING $\preceq_\ell$.

$$\pi_1 \preceq_\ell \pi_2 \iff (\exists \mathbf{x} \in \pi_1)(\exists \mathbf{y} \in \pi_2)(\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}) .$$

The local product ordering $\preceq_\ell$ is reflexive, but not antisymmetric (that is why we write $\pi_1 -_\ell \pi_2$ if both $\pi_1 \preceq_\ell \pi_2$ and $\pi_2 \preceq_\ell \pi_1$) nor transitive, and $\prec_\ell$ may contain cycles.

Consider two objects $a, b \in \Omega$ and a partition Π of $\mathcal{X}$. The **partial dominance relation (w.r.t.** Π**)** $\trianglelefteq_p$ (see p. 166) is defined by

PARTIAL DOMINANCE RELATION (W.R.T. Π) $\trianglelefteq_p$.

$$a \trianglelefteq_p b \iff \pi_1 \lesssim_\ell \pi_2 ,$$

where $\mathbf{a} \in \pi_1$ and $b \in \pi_2$. When all the blocks of Π consist of singletons, this relation is just the **dominance relation** $\trianglelefteq$ (see p. 98). Remark that $a \trianglelefteq b$ if and only if $\mathbf{a} \leq_{\mathcal{X}} b$.

DOMINANCE RELATION $\trianglelefteq$.

7.1.2 Validation versus modelling

Validation. The term "validation" only refers to finished models, i.e. input-output boxes, whether they are black or white. It refers to questions like: What is the accuracy of the model? How acceptable is the model? How useful is the model? Some of these properties can be measured quantitatively by some **risk functional** (see p. 116) (also called **error function**) R. Other properties refer to some demands that can only be described qualitatively. For example, many situations are conceivable where a black box model is not considered as useful, even if its accuracy on some test set is the full 100%. But also a rule-based model containing contradicting rules could be deemed unacceptable for some people.

RISK FUNCTIONAL.

[1]The data space $\mathcal{X}$ itself induces a partition of Ω.

Modelling. In general, validation is an important aspect of modelling: in most techniques for modelling a classification, the risk functional R used for validation is somehow incorporated, either directly or indirectly. For example in the k-Nearest Neighbour method, the parameter k can be estimated to minimise the expected error R using a leave-one-out cross validation on the learning sample; the pruning of a tree is all about maintaining or even decreasing the expected error while simultaneously decreasing the complexity of the model; the whole theory of support vector machines is based on what is called the structural minimisation of the risk functional (structural risk minimisation [54, 115]).

Still, despite the importance of validation, it remains but *one* aspect of the modelling process. Indeed, essentially, the modelling process is about finding one adequate model among infinitely many possible models. Even if the space of possible models is restricted to a certain family of models, e.g. models that can be represented by a decision tree, then it is in most situations still impossible to do an exhaustive search to find the "optimal" model(s). Therefore, we need guidelines to help us see the wood for the trees.

Validation and ranking. Our point of departure is that we are confronted with the problem of learning a ranking based on a fixed set of criteria. This means that in general a model for this ranking should be monotone to be acceptable[2]. In view of such a demand, any non-monotone model should immediately be disregarded, even if its accuracy on some test set is the full 100%.

Assumption: the user demands the monotonicity of the model.

This means that the representation $(\lambda_{\text{repr}}, \trianglelefteq_{\text{Im}})$ induced by the model must be monotone, i.e. for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ it must hold that

$$\boxed{\mathbf{x} \leq_{\mathcal{X}} \mathbf{y} \Rightarrow \lambda_{\text{repr}}(\mathbf{x}) \trianglelefteq_{\text{Im}} \lambda_{\text{repr}}(\mathbf{y})}$$

In Chapter 4, we showed how simple set-based and distribution-based representations λ_{repr} could be made monotone. And in Section 6.5.4 (see p. 173), we showed that we do not need to alter these representation on a point by point basis to achieve this, but that the same feat can be accomplished on the level of any partition embedded in the partition induced by λ_{repr} (of course, as a downside, this latter process is usually much coarser if not carried out with care).

[2]This was discussed in Section 4.5.3, in particular in the paragraph entitled *About doubt and reversed preference* (see p. 100). Of course, in some situations, the user can live with some minor degree of non-monotonicity, e.g. if there is only a 5% chance that there is rank reversal of order 1 between two objects, a 0.5% chance the reversal is of order 2, and in no situation, the reversal is of order higher than 2, where the order of rank reversal could be defined as the rank difference of two objects leading to reversed preference (for example on the ordinal scale Bad, Moderate, Good, Very Good, we have rank reversal of order 1 if a Good object is ranked lower than a Moderate object, and of order 2 if a Very Good object is ranked lower than a Moderate object).

Modelling and ranking. While the major part of Chapter 4 is mainly a contribution to the validation process, the bigger part of Chapter 6 deals directly with a guideline for model building when dealing with partition-based modelling: the *principle of partial dominance preservation* (see p. 167):

$$a \lhd_p b \Rightarrow \lambda(a) \leq_{\mathcal{L}} \lambda(b)\,.$$

Although this principle can not always be fully satisfied during the modelling phase because of the restrictions met during the modelling (as we will see below), we would like it to be fulfilled as much as possible. In other words, it gives us the guideline to reduce violations against this principle (i.e. the **reversed preferences** (see p. 100)) to a minimum.

REVERSED PREFERENCES.

Broad guideline: minimise reversed preferences.

7.1.3 Monotone and non-monotone data sets

In this section, we take a closer look at the problems in finding a monotone model starting from monotone and non-monotone learning samples.

Monotone data sets. A data set $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$ is called **monotone** if

MONOTONE DATA SET.

$$\mathbf{a} \leq_{\mathcal{X}} \mathbf{b} \Rightarrow d(a) \leq_{\mathcal{L}} d(b)\,.$$

Now assume we are confronted with such a monotone learning sample $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$, and our mission is to build a monotone partition-based model $(\lambda^*_{\text{repr}}, \leq_{\mathcal{L}})$ such that for all $a \in \mathcal{S}$, we have $\lambda^*_{\text{repr}}(a) = d(a)$. This problem can be solved by finding some partition Π of $\mathcal{X}$ such that for all $a, b \in \mathcal{S}$ it holds that

$$\Pi(\mathbf{a}) \lesssim_{\ell} \Pi(\mathbf{b}) \Rightarrow d(a) \leq_{\mathcal{L}} d(b)\,. \tag{7.1.1}$$

Indeed, if this demand is satisfied, we have in particular that

$$\Pi(\mathbf{a}) \sim_{\ell} \Pi(\mathbf{b}) \Rightarrow d(a) = d(b)\,,$$

whence also $\Pi(\mathbf{a}) = \Pi(\mathbf{b}) \Rightarrow d(a) = d(b)$. Because of this, we can define the representation λ^*_{repr} as the function that maps all elements in the same block $\pi_\Omega \in \Pi_\Omega$ (with $\pi_\Omega \cap \mathcal{S} \neq \emptyset$) onto the single value of the samples it contains. By definition, we have that $(\lambda^*_{\text{repr}}, \leq_{\mathcal{L}})$ is monotone on this subset of objects and that λ^*_{repr} coincides with d on $\mathcal{S}$. The other blocks may be labelled at will as long as the monotonicity of $(\lambda^*_{\text{repr}}, \leq_{\mathcal{L}})$ is safeguarded (which of course is always possible). Remark that, as the data set is monotone, we know there exists at least one such a partition, namely the trivial partition where each vector of $\mathcal{X}$ is a block.

Non-monotone data sets.

WHY? The definition of monotone data sets is rather limiting, because it does not allow for any uncertainty, not even the simple occurrence of doubt within the data. It strictly prevents the occurrence of two objects $a, b \in \Omega$ with the same description $\mathbf{a} = \mathbf{b}$, but having different labels $d(a) \neq d(b)$. Moreover, monotone data sets exclude the idea of error, except if by coincidence, it would comply with the monotonicity demand. Lastly, monotone data sets prohibit any occurrence of reversed preference, even the ones that are not due to errors. This happens for example in surveys where each person is asked to give partial evaluations as well as a global evaluation (either about one subject fixed for everybody, e.g. student evaluation of lecturers, or about a subject linked to their personal situation, e.g. noise annoyance). Clearly, nothing guarantees these responses will not contradict each other.

HOW? INVASIVE APPROACHES. There are several possible solutions to deal with the more realistic case of non-monotone learning samples. One is to define a new set of criteria C such that the new adapted data set becomes monotone, another is to simply remove the examples causing trouble. This latter approach is the most frequently used, e.g. [74, 117], and is in fact the essence of the method OLM [9], see also Section 3.1.1 (see p. 53). However, there lurk some immediate dangers in removing examples from the learning sample in order to make it monotone, the same problem that appears in data cleansing: How can you determine what are the erroneous examples to be removed? And for that matter: How can you determine that the non-monotonicity was caused by errors in the first place? If you remove the correct example, and leave in the error, the relative number of incorrect examples in the data set becomes higher, while the total number of examples in the data set itself diminishes. And if the problem was not caused by an error in the first place, then you have deliberately ignored potentially useful information. In other words, before an example is removed, there should better be some good reasons to do so (e.g. a decent technique for identifying outliers in a ranking problem[3]). Finally, the MDT algorithm (Monotone Decision Trees [89]) was recently adapted to be able to deal with non-monotone data [17, 18, 87]. This method is however based on a technique of adding generated data and relabelling existing data. We believe that the same caution as with removing objects should be administered to adding non-authentic data, or altering the original examples.

HOW? A NON-INVASIVE APPROACH. We believe that in dealing with non-monotone data sets, we should not invade into the learning sample itself, incorporating the information of every example as it is into the final result. This brings us to the following simple idea for partition-based models: try to find a partition Π such that (7.1.1) holds except for the objects $a, b \in \mathcal{S}$ for which

$$(a \trianglelefteq b) \wedge (d(a) \not\leq_{\mathcal{L}} d(b)) \,,$$

since these occurrences of doubt and reversed preferences are inherent to the given problem, and can never be eliminated. We also say they are **unsolvable**.

SOLVABLE DOUBT/REVERSED PREFERENCE.

[3]Note that such a technique does not yet exist (at least not at the moment of this writing).

As was the case for monotone data sets, the trivial partition of $\mathcal{X}$ always complies to this demand. So, we can refine our previous guideline to

$$\boxed{\text{Broad guideline: minimise solvable reversed preferences.}}$$

7.2 Principles of growing

Notions and conventions. We assume the reader is familiar with the basic ideas of growing (splitting) and pruning a tree. A short introduction can be found in Appendix 1.A (see p. 14).

NODES. Every node t of a tree T corresponds to some subset of the data space $\mathcal{X}$, which we will denote by $t_{\mathcal{X}}$. As a consequence, t also corresponds to some subset t_Ω of the object space Ω, i.e. $t_\Omega = \rho^{-1}(t_{\mathcal{X}})$, where $\rho : \Omega \to \mathcal{X}; a \mapsto (c_1(a), \ldots, c_n(a))$. In a less formal way, we sometimes simply write $a \in t$ or $\mathbf{x} \in t$ instead of $a \in t_\Omega$ and $\mathbf{x} \in t_{\mathcal{X}}$. A node of a tree T is called a **leaf** if it has no children, otherwise it is called **inner node**. The set of leaves of T is denoted by $\widetilde{T}$. Let t be an inner node of T. The subtree of T starting at t is denoted by T_t. See also Figure 7.1.

$t_{\mathcal{X}}$
t_Ω
LEAF.
INNER NODE.
$\widetilde{T}, T_t$

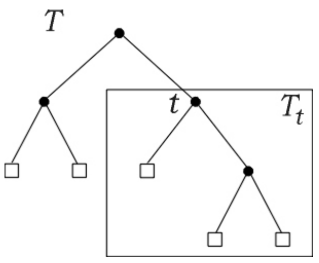


Figure 7.1: The subtree T_t of T. Leaves are denoted by $\square$, inner nodes by $\bullet$.

SPLITS. Performing a single split s on a tree T involves identifying the leaf t to be split, and some test h on the objects from the leaf t which determines in which child of t the object will fall. If the split s is binary, the test has a yes-no response, objects that answer positive are sent to the left child, the others to the right child. Assuming we are only dealing with true criteria, a **single univariate binary split** s on a tree T is a triplet (t, c, v), where

SINGLE UNIVARIATE BINARY SPLIT $s = (t, c, v)$.

- $t \in \widetilde{T}$ is a leaf of T, namely the leaf to be split,
- $c \in C$ is a criterion on which objects from t will be tested
- $v \in \mathcal{X}_c$ is a value from the image of the criterion c. For any $a \in t$, if $c(a) \leq_c v$ then $a \in t_L$, otherwise $a \in t_R$, where t_L (resp. t_R) denotes the left (resp. right) child of t after the split is carried out.

t_L, t_R

If the leaf to be split is clear from the context, we write $s = (c, v)$, or sometimes even simply $c \leq_c v$. In this thesis, we only consider univariate splits.

$T(s)$

T/t

Let s be a split on T. We denote the tree obtained from T after splitting it according to s by $T(s)$. Let t be an inner node of T. Pruning [23] a branch T_t from a tree T consists of deleting from T all descendants of t, that is, cutting off all of T_t except its root node. We denote the resulting pruned tree by T/t. See also Figure 7.2.

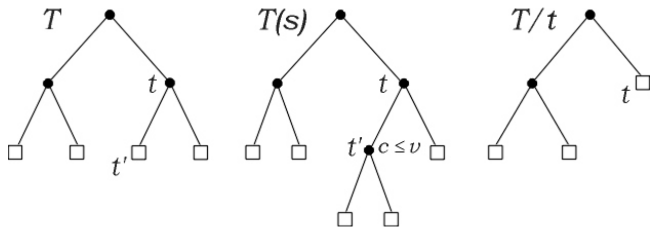


Figure 7.2: Splitting and pruning: T, $T(s)$ with $s = (t, c, v)$ and T/t.

TWOING CRITERION.

SPLITTING RULE. There exist many kinds of different splitting rules (also called splitting measure or splitting criterion). In this chapter, we only mention the **twoing criterion**, adapted to ordinal class labels [23]. Its value for a binary split s is calculated as follows. For each $i \in \mathcal{L}$, relabel the objects in the children of t as belonging to a class $\leq_{\mathcal{L}} i$, or belonging to a class $>_{\mathcal{L}} i$. Now calculate the Gini diversity indices G_i on the split s based on these new labels, i.e. calculate

$$G_i = p(t_L)\, p(\leq_{\mathcal{L}} i|t_L)\, p(>_{\mathcal{L}} i|t_L) + p(t_R)\, p(\leq_{\mathcal{L}} i|t_R)\, p(>_{\mathcal{L}} i|t_R)\,,$$

where $p(t)$ is the estimated probability that an object falls into the node t, and $p(j|t)$ is the estimated probability that an object belongs to class j if it is know to fall into t. The value of the twoing criterion is just the minimum of all G_i.

7.2.1 Ordering the leaves

Leaves and intervals. A tree is nothing but a nice representation of a family of partitions that has some nice properties. For example, it was shown in [89] that all blocks of a partition induced by a tree are intervals in $(\mathcal{X}, \leq_{\mathcal{X}})$:

Lemma 7.2.1. [89] *Let T be a tree. For all leaves $t \in \widetilde{T}$ it holds that*

$$\boxed{t_{\mathcal{X}} = [\mathbf{x}, \mathbf{y}]}$$

for some $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ with $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}$.

We will adhere to the following notations as used in [89]: the minimal and maximal element in $\mathcal{X}$ that characterise the interval induced by a leaf t are denoted by $a(t)$ and $b(t)$, i.e. $t_{\mathcal{X}} = [a(t), b(t)]$.

An order on the leaves. Since the leaves can be identified with blocks of a partition, we can use the local product preorder to order them.

Definition 7.2.1

Let T be a tree. We define the order $\leq_T$ on the leaves of T as follows: let $t, t' \in \widetilde{T}$, then

$$t \leq_T t' \iff t_{\mathcal{X}} \lesssim_\ell t'_{\mathcal{X}} \,.$$

We also write

$$t \preceq_T t' \iff t_{\mathcal{X}} \preceq_\ell t'_{\mathcal{X}} \,.$$

We already noted that all the blocks of the partitions induced by a tree are intervals in $\mathcal{X}$. The following lemma tells us how the local product ordering $\preceq_\ell$ behaves on intervals:

Lemma 7.2.2. *Consider two non-empty intervals* $[\mathbf{x}, \mathbf{y}]$ *and* $[\mathbf{x}', \mathbf{y}']$ *in* $(\mathcal{X}, \leq_{\mathcal{X}})$, *then it holds that*

(i) $[\mathbf{x}, \mathbf{y}] \preceq_\ell [\mathbf{x}', \mathbf{y}'] \iff (\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}')$

(ii) $[\mathbf{x}, \mathbf{y}] -_\ell [\mathbf{x}', \mathbf{y}'] \iff [\mathbf{x}, \mathbf{y}] \cap [\mathbf{x}', \mathbf{y}'] \neq \emptyset$

Proof.
(i) Assume $[\mathbf{x}, \mathbf{y}] \preceq_\ell [\mathbf{x}', \mathbf{y}']$. Then $(\exists \mathbf{z} \in [\mathbf{x}, \mathbf{y}])(\exists \mathbf{z}' \in [\mathbf{x}', \mathbf{y}'])(\mathbf{z} \leq_{\mathcal{X}} \mathbf{z}')$. So, $\mathbf{x} \leq_{\mathcal{X}} \mathbf{z} \leq_{\mathcal{X}} \mathbf{z}' \leq_{\mathcal{X}} \mathbf{y}'$. The converse holds by definition.

(ii) Assume $[\mathbf{x}, \mathbf{y}] -_\ell [\mathbf{x}', \mathbf{y}']$. Then (i) learns us that we have both $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}'$ and $\mathbf{y} \leq_{\mathcal{X}} \mathbf{x}'$, i.e. $(\forall i \in N)(x_i \leq_{c_i} y'_i)$ and $(\forall i \in N)(x'_i \leq_{c_i} y_i)$. Let $K_1 = \{i \in N \mid y_i <_{c_i} y'_i\}$, and define the vector $\mathbf{z}$ by

$$z_i = \begin{cases} y_i & \text{, if } i \in K_1, \\ y'_i & \text{, otherwise.} \end{cases}$$

We have for all $i \in K_1$ that $x_i \leq_{c_i} z_i = y_i$ because $[\mathbf{x}, \mathbf{y}]$ is a non-empty interval; $x'_i \leq_{c_i} z_i = y_i$ because $\mathbf{x}' \leq_{\mathcal{X}} \mathbf{y}$; $z_i = y_i \leq_{c_i} y_i$; and $z_i = y_i \leq_{c_i} y'_i$ by definition of K_2.
For all $i \in N \setminus K_1$, we find $x_i \leq_{c_i} z_i = y'_i$ because $\mathbf{x} \leq_{\mathcal{X}} \mathbf{y}'$; $x'_i \leq_{c_i} z_i = y'_i$ because $[\mathbf{x}', \mathbf{y}']$ is a non-empty interval; $z_i = y'_i \leq_{c_i} y_i$ because $i \notin K_1$; $z_i = y'_i \leq_{c_i} y'_i$.
In all, we find that $\mathbf{z} \in [\mathbf{x}, \mathbf{y}] \cap [\mathbf{x}', \mathbf{y}']$.
Conversely, if $\mathbf{z} \in [\mathbf{x}, \mathbf{y}] \cap [\mathbf{x}', \mathbf{y}']$, then $\mathbf{x} \leq_{\mathcal{X}} \mathbf{z} \leq_{\mathcal{X}} \mathbf{y}'$, and $\mathbf{x}' \leq_{\mathcal{X}} \mathbf{z} \leq_{\mathcal{X}} \mathbf{y}$. □

As a consequence we have that

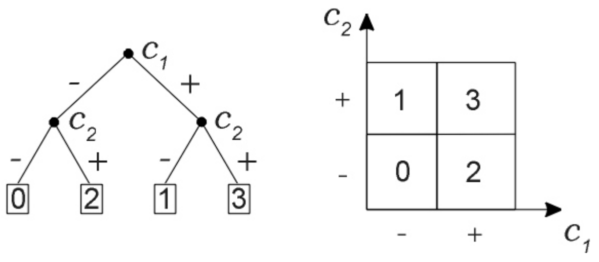
$$\boxed{t \preceq_T t' \iff a(t) \leq_{\mathcal{X}} b(t')}$$

and that the ordering $\preceq_T$ is antisymmetrical: for any two leaves $t, t' \in \widetilde{T}$, we always have $t_{\mathcal{X}} \cap t'_{\mathcal{X}} = \emptyset$, except if $t = t'$ (because the leaves of a tree induce a partition of $\mathcal{X}$). This makes that $\leq_T$ becomes an order, i.e. reflexive, antisymmetrical and transitive! Moreover, it can be shown that $\prec_T$ and $<_T$ contain no cycles, the proof of which is deferred to Section 7.6, Proposition 7.6.2 and Corollary 7.6.3 (see p. 216).

Proposition 7.2.3. *The relations $\prec_T$ and $<_T$ contain no cycles.*

7.2.2 Visualisation

One of the key aspects of decision trees is their easily interpretable graphical representation, reflecting all information at a single glance. However, for ranking problems, the basic tree visualisation is hiding the essential ordering information. For example, the tree in Figure 7.3(a) is monotone, as can be easily seen from the



(a) Tree. (b) Induced partition.

Figure 7.3: A monotone tree.

partition it induces (figure (b)). However, the monotonicity is not immediately obvious from the pictured tree itself. If the tree gets more complex, the problem is even more imposing. Even the visualisation of the induced partition will bring little comfort, as for dimensions higher than two, the partition becomes unreadable. But all is not lost, the following lemma comes to the rescue!

Lemma 7.2.4. *The poset $(\widetilde{T}, \leq_T)$ is a lattice.*

Proof.
We only need to prove that for each pair of leaves t_1 and t_2 in $\widetilde{T}$ there exist a greatest lower bound and a least upper bound.
Consider $\mathbf{x} = \inf\{b(t_1), b(t_2)\}$, then $\mathbf{x} \in \mathcal{X}$ because $(\mathcal{X}, \leq_{\mathcal{X}})$ is a lattice, and hence there is a unique t^* such that $\mathbf{x} \in t^*_{\mathcal{X}}$. Clearly $t^* \leq_T t_1$ (because $a(t^*) \leq_{\mathcal{X}} \mathbf{x} \leq_{\mathcal{X}} b(t_1)$) and $t^* \leq_T t_2$, so t^* is a lower bound of t_1 and t_2. Moreover, it is the greatest lower bound. Indeed, assume t' is another lower bound of t_1 and t_2, implying that $a(t') \leq_{\mathcal{X}} b(t_1)$ and $a(t') \leq_{\mathcal{X}} b(t_2)$. Because $(\mathcal{X}, \leq_{\mathcal{X}})$ is a lattice, we may derive from this that $a(t') \leq_{\mathcal{X}} \mathbf{x} \leq_{\mathcal{X}} b(t^*)$, or $t' \leq_T t^*$.
Analogous for the least upper bound. □

So, we can always draw a lattice reflecting the order $\leq_T$, where the nodes of the lattice correspond to the leaves of the tree. If we tilt the lattice 90 degrees clockwise (i.e. the infimum on the left instead of on the bottom, and the supremum on the right instead of on the top), we can always – provided that we play a bit around with the different possibilities for positioning – draw nice diagrams like in Figure 7.4. We

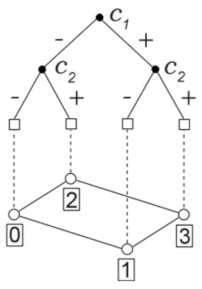


Figure 7.4: A monotone tree T with dominance graph $(\widetilde{T}, \leq_T)$.

will call such a graph a **dominance graph**[4].

DOMINANCE GRAPH.

7.2.3 Growing

Introduction. It is already known quite some time that the process of building a tree is best done in two steps: first grow an overly large tree to obtain a near-to-perfect fit of the training data, then prune the branches that seem[5] to cause an overfitting of the problem. This section handles about the growing phase of a tree for a ranking problem.

A naive algorithm. In Section 7.1.3 we mentioned that the goal is to obtain a partition that fulfils the demand (7.1.1) except for unsolvable doubt and reversed preference, or translated to the context of decision trees:

Stopping condition for splitting: when for all $a \in t_\Omega$ and for all $b \in t'_\Omega$ it holds that

$$t \leq_T t' \Rightarrow d(a) \leq_{\mathcal{L}} d(b)\,,$$

except if the pair (a, b) leads to doubt or reversed preference in $(d, \leq_{\mathcal{L}})$.

This is the equivalent of demanding that all leaves are pure or as pure as possible in a regular classification context.

[4]Remark that this dominance graph is nothing else but the locally compressed graph mentioned in Section 6.3.3, paragraph "Why local?" (see p. 163).

[5]According to [24, 98], pruning does not necessarily lead to better predictive accuracy. However, it does lead to less complex and hence easier comprehensible trees.

We can now derive a naive[6] monotone tree algorithm: just keep on splitting until the previous demand is met. Remark that it is very probable that we will obtain empty leaves, i.e. leaves that contain no objects from $\mathcal{S}$, in order to satisfy this demand. In that case, the problem becomes: How do we label these empty leaves? A more sophisticated, but still a bit naive[7] tree growing approach was proposed in [74, 90].

A less naive algorithm. The broad guideline mentioned in Section 7.1.3 helps us in finding a more adapted scheme: for each split, choose the one that minimises the number of reversed preferences between $(d, \leq_{\mathcal{L}})$ and $\trianglelefteq_p$. Because our second objective is to minimise impurity[8] in the tree, we can for example break ties using the twoing criterion, which tends to choose more balanced splits (see [23]). This idea could be said to be more sophisticated in its dealing with reversed preference, while rather naive in its dealing with doubt in case there is reversed preference. Still, it should be remarked that this is a good starting point, since minimising reversed preference will automatically reduce doubt/impurity[9], while reducing doubt/impurity may have an increasing effect on reversed preference. Of course, the best would be to have some kind of measure that can incorporate both doubt and reversed preference on the same level[10].

About overfitting. In the case of non-monotone data, the proposed approach is likely to overfit the data, just because of the reversed preferences inherent to the given learning sample. In Chapter 4 and Section 6.5.3 (see p. 170), we already showed how reversed preference leads to additional doubt in the final classifier by the idea of turning reversed preference into doubt to assure monotonicity. Indeed, consider the non-monotone learning sample $(\mathcal{S}, (d, \leq_{\mathcal{L}}))$ with only one criterion, i.e. $\mathcal{X} = \mathcal{X}_c$ is one-dimensional, as given in Table 7.1. From d, we can define a consistent representation $(\lambda_{\text{repr}}, \leq^{[2]})$ as shown in the same table. However, if we want to grow a tree on this data set, and only stop growing when the above stopping condition is met, we see that we have to separate the objects a_3 and a_4 into different leaves because they do not lead to doubt nor to reversed preference. So the minimal tree that satisfies the stopping condition has four leaves t^1, t^2, t^3, t^4, with $t^1_{\mathcal{X}} = [1,1]$, $t^2_{\mathcal{X}} = [2,3]$, $t^3_{\mathcal{X}} = [4,5]$ and $t^4_{\mathcal{X}} = [6,6]$ for example the tree depicted alongside the table. Obviously, if the node t would be pruned, merging the leaves t^2 and t^3, we would obtain a tree with the same performance.

About the interdependency of the leaves. There is another very important remark to be made. In classification trees, all leaves are independent. This makes

[6]Naive in the sense that we can always grow a perfect tree if we just keep on splitting long enough.

[7]Sophisticated in its dealing with doubt and in its labelling, but still a bit naive in its dealing with reversed preference. See also Appendix 3.A where we solidify this statement.

[8]A set is pure if it contains only objects with the same label, the more diverse the content of the set, the more impure it becomes. Impurity can be measured by e.g. the Gini diversity index or the Shannon entropy [23], see also Section 2.4.2, p. 39.

[9]Doubt and impurity are closely related as demonstrated in Section 2.5.3 (see p. 46).

[10]We elaborate on a first proposal of such a measure in [27].

(a) Table.

	a_1	a_2	a_3	a_4	a_5	a_6
c	1	2	3	4	5	6
d	1	2	1	2	1	2
λ_{repr}	1	[1,2]	[1,2]	[1,2]	[1,2]	2
	t^1	t^2		t^3		t^4

(b) Tree.

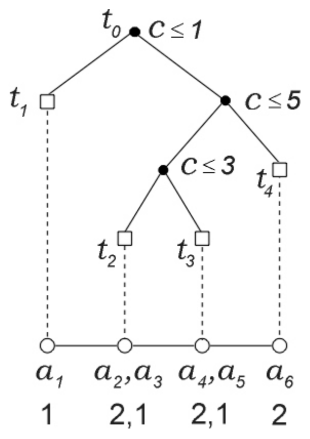


Table 7.1: A simple learning sample $(d, \leq_{\mathcal{L}})$ and a monotone model.

that each node can be seen as the root of a new classification tree that can be dealt with on its own. However, in the ranking problem, all leaves are interconnected by the order $\leq_T$ and must therefore always be considered as a whole. This means that while for classification trees, the local optimum of the splitting measure at a node coincides with the global optimum for the whole tree, this is not necessarily true for ranking trees. So besides the local component, ranking trees must always keep a kind of global control on what is happening during splitting, because the split of one leaf may have an effect on all the other leaves. As a consequence, the order in which the leaves are split starts playing a distinctive role. For now, we will follow a "breadth first" strategy, or even better, an "impurest first" strategy (meaning that the node with the highest value on some impurity measure weighted by the node's weight is chosen). At the end of Section 7.3.3 we will discuss how only considering splits of one leaf at a time may be inappropriate for finding a good solution, and in Section 7.5.1 we will suggest yet another scheme for picking the next nodes to consider for splitting.

Another complexity problem. For each possible split, in order to calculate some measure on the leaves, we first need to determine the order $\leq_T$ on the leaves. The definition of $\leq_T$ involves a transitive closure of another relation, and this may be very time-consuming. In Section 7.6, we discuss a way of generating the relation $\leq_T$ after the split from the relation before the split with minimal additional calculations.

7.2.4 Summary

Every leaf t of a tree T corresponds to an interval $[a(t), b(t)]$ in $\mathcal{X}$. We can define a partial order $\leq_T$ on the set of leaves $\widetilde{T}$ of T in two steps:

(i) Define the reflexive and antisymmetric relation $\preceq_T$ by:

$$t \preceq_T t' \iff a(t) \leq_{\mathcal{X}} b(t')\,.$$

(ii) Consider the transitive closure $\leq_T$ of $\preceq_T$.

The resulting poset $(\widetilde{T}, \leq_T)$ is a lattice, which enables us to easily draw the *dominance graph*.

While splitting, the main guideline is to minimise the occurrence of solvable reversed preference. T is grown until the stopping condition is met, i.e. until for all $t, t' \in \widetilde{T}$ it holds that

$$(\forall a \in t_\Omega)(\forall b \in t'_\Omega)(t \leq_T t' \Rightarrow d(a) \leq_{\mathcal{L}} d(b))\,,$$

except for the pairs (a, b) leading to doubt or reversed preference within the learning sample.

7.3 Examples of growing

7.3.1 Minimal example, monotone data set

Candidate evaluations. Consider again the candidate evaluation example of Section 1.2.2 (see p. 6), as depicted in Table 7.2. Candidates are evaluated according

	c_1	c_2	c_3	d
a_1	$-$	$-$	$+$	B
a_2	$+$	$-$	$-$	M
a_3	$-$	$+$	$+$	G
a_4	$+$	$+$	$-$	M

Table 7.2: Evaluations of candidates

to c_1, their capacity for learning (slow or fast), c_2, their working experience (little or much), and c_3, their personal profile, i.e. how well they will fit into the group they have to work with (bad or good). These binary values are denoted by $-$ and $+$. The set of labels is $\mathcal{L} = \{\mathsf{B}(\text{ad}) <_{\mathcal{L}} \mathsf{M}(\text{oderate}) <_{\mathcal{L}} \mathsf{G}(\text{ood})\}$. We have seen in Section 1.2.2 that a classification tree algorithm run on this data set leads to a non-monotone tree, although this learning sample is clearly monotone.

Growing the tree. Let us now demonstrate the principles of the previous section. There are three possibilities for the first split. These are pictured together with their dominance graph in Figure 7.5. We see that the first option results in 2 occurrences

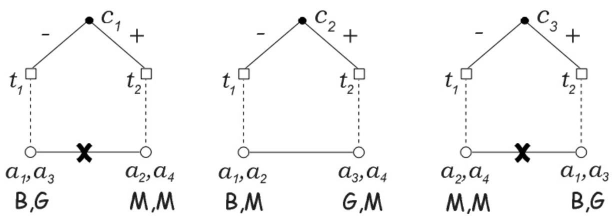


Figure 7.5: The splits based on c_1, c_2 and c_3, along with the corresponding dominance graph. We mark the occurrence of reversed preference with a cross

of reversed preference: $a_3 \in t_1 \leq_T t_2 \ni a_2, a_4$, while $d(a_3) = \mathsf{G} \not\leq_{\mathcal{L}} d(a_2) = d(a_4) = \mathsf{M}$. The split based on c_2, however, is in accordance with the principle of partial dominance preservation: $(\forall a \in t)(\forall b \in t')(t \leq_T t' \Rightarrow d(a) \leq_{\mathcal{L}} d(b))$. The third possible split leads again twice to reversed preference. So we obtain $c_2 > c_1 = c_3$ as the ranking of the possible splits.
Now that the first split has been chosen, we look at the possibilities for the second split as depicted in Figure 7.6. We see clearly that the split based on c_3 leads again

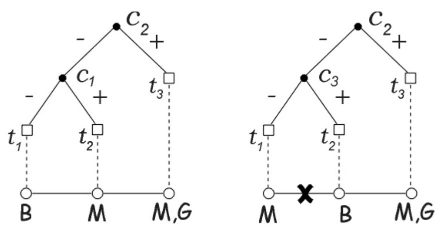


Figure 7.6: Two possibilities for the second split.

to reversed preference: $a_2 \in t_1 \leq_T t_2 \ni a_1$ and $d(a_2) = \mathsf{M} \nleq_{\mathcal{L}} d(a_1) = \mathsf{B}$. The split based on criterion c_1 is in line with the principle and is therefore chosen. Finally, the last split is best based on c_3 as can be seen in Figure 7.7.

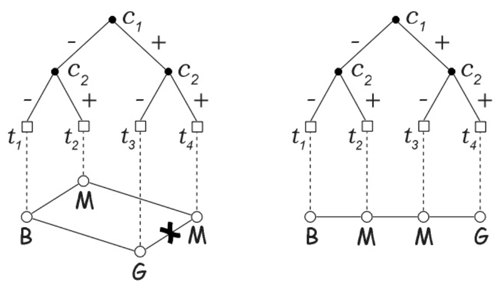


Figure 7.7: Two possibilities for the third split.

The rule base. In this example, all leaves contain at least one example, so we are not confronted with the labelling of empty leaves. We end up with the following rule base:

- if the candidate has little or no working experience and if, moreover, (s)he is a slow learner, then (s)he gets the global evaluation *Bad*,
- if the candidate has little or no working experience but can compensate this a bit by being a fast learner, then (s)he is evaluated *Moderate*,
- if the candidate has a lot of working experience, but doesn't fit well into the group, then (s)he is evaluated *Moderate*,
- if the candidate has a lot of working experience combined with a good fit into the group, then (s)he is evaluated *Good*.

This rule base is very natural, especially compared to the rule bases induced from a classification tree algorithm (see Section 1.2.2, p. 7).

7.3.2 Minimal example, non-monotone data set

Consider the example from Section 4.6.3 (see p. 105), retaken in Table 7.3.

	a_1	a_2	a_3	a_4	a_5	a_6
c	2	1	4	3	6	5
d	1	2	3	4	5	6

Table 7.3: A simple non-monotone ranking $(d, \leq_{\mathcal{L}})$.

Here, we are confronted with binary splits of the form $c \leq_c v$. For the first split, we see that the splits $c \leq_c i$ with $i = 1, 3, 5$ lead to reversed preference, while the splits $c \leq_c j$ with $j = 2, 4$ do not (see Figure 7.8). Since these latter splits

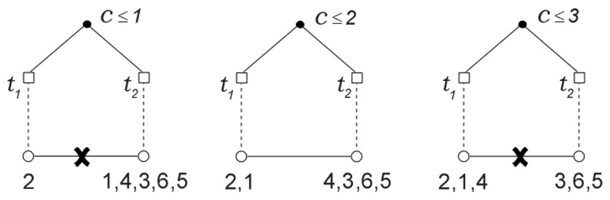
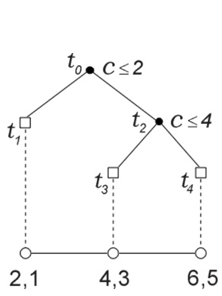


Figure 7.8: Some possibilities for the first split.

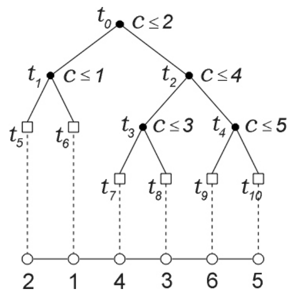
also lead to the same value of the twoing criterion, we simply choose the first one, $c \leq_c 2$, as our first split. If we now consider an "impurest first" strategy for picking the next node to split, we come up with node t_2. A "breadth first" strategy would lead to t_1. We now remark that anyway, t_1 only contains two objects that lead to reversed preference in the learning sample, so splitting t_1 doesn't get us anywhere. For the split of t_2, there are three possibilities left: $c \leq 3$ leading to reversed preference, $c \leq 4$ which is consistent, and $c \leq 5$ leading again to reversed preference. Therefore $c \leq 4$ is chosen and we end up with the final tree complying with our stopping demand, as depicted in Figure 7.9 (a).

But... There is however an important "But...". In the previous case, it was easy to detect that the splitting of t_1 is useless. But is this always true? The answer is unfortunately negative[11]. Therefore, we only stop considering a node for splitting in case it is pure and if it does not lead to solvable reversed preferences with other leaves. Moreover, while splitting, we check whether the best split does at least something: either removing doubt or removing solvable reversed preferences. If not, the split is not carried out. This scheme would lead to the final tree shown in Figure 7.9 (b). Clearly, this tree is in need of some pruning!

[11]Maybe there exists a positive answer to this question, but we haven't yet figured that one out.



(a) Final tree T_1.



(b) Final tree T_2.

Figure 7.9: Final trees for Table 7.3.

7.3.3 A more involved example

Introduction The examples elaborated in the previous section are extremely well behaving: we can always find a split that causes no reversed preference (at least not solvable ones). This implies that the resulting children can be considered as independent, because meddling with them can not affect the other leaves (influence on other leaves can only be achieved via reversed preferences). The next example taken from [48], which is an adaptation (a recoding) of a data set in [31], does not have this friendly property.

The contraception data. The data are shown in Table 7.3.3, where $0 =$ low, $1 =$ moderate, $2 =$ high. The global label given by d stands for the "use of contraception", the partial evaluations are based on the following criteria:

- Average years of education (c_1),
- Urbanisation (c_2),
- Gross national product per capita (c_3),
- Expenditure on family planning (c_4).

Remark that this data set is not monotone, although nearly. The countries Sri Lanka (4) and Thailand (6) cause reversed preference in the learning sample. As mentioned above, we could eliminate one of them, but in this case, this would mean disregarding information because all data is correct. The reason for the reversed preference is likely to be caused by a too small set of criteria (see also Section 4.6.2, p. 101).

Growing a ranking tree There are eight possible splits to start with: $c_i \le 0$ and $c_i \le 1$ for $i = 1, \dots, 4$. Some of them are shown in Figure 7.10, a table with the number of solvable reversed preferences corresponding to all splits can be found

Country	c_1	c_2	c_3	c_4	d
(1) Lesotho	1	0	0	0	0
(2) Kenya	0	0	0	0	0
(3) Peru	1	1	2	0	0
(4) Sri Lanka	1	1	0	1	0
(5) Indonesia	0	0	0	1	0
(6) Thailand	1	0	0	1	1
(7) Colombia	1	2	1	1	1
(8) Malaysia	0	1	2	1	1
(9) Guyana	2	1	2	0	1
(10) Jamaica	2	0	2	2	1
(11) Jordan	0	2	1	0	1
(12) Panama	2	2	2	1	2
(13) Costa Rica	2	1	2	2	2
(14) Fiji	1	1	2	2	2
(15) Korea	2	1	1	2	2

c_1	: Average years of education
c_2	: Urbanisation
c_3	: Gross national product per capita
c_4	: Expenditure on family planning
d	: Use of contraception
0=low, 1=moderate, 2=high	

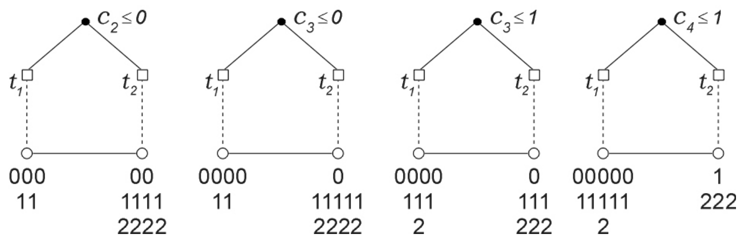
Table 7.4: Recoded contraception data

on the bottom of this figure. Consider for example the split $c_2 \leq 0$. It contains four reversed preferences: two objects with label 1 belong to t_1 and two objects with label 0 belong to t_2, while $t_1 <_T t_2$. However, one of these pairs causing reversed preference is the pair (Sri Lanka, Thailand), which is an unsolvable one. Therefore we end up with three solvable reversed preferences for the split $c_2 \leq 0$. In all, we see we have a tie between $c_3 \leq 0$ and $c_4 \leq 1$. This tie is broken by calculating that the split $c_4 \leq 1$ has the lowest value for the twoing measure.

Now, following the impurest first strategy, we continue splitting the node t_1. The splits are shown in Figure 7.11. This time, we have to consider three leaves in our calculations, for example for the split $c_1 \leq 0$, we find six reversed preferences between t_3 and t_4, one between t_4 and t_2, and none between t_3 and t_2. Considering all splits, we find again a tie, now between $c_1 \leq 1$ and $c_2 \leq 1$. The matter is settled in favour of $c_1 \leq 1$.

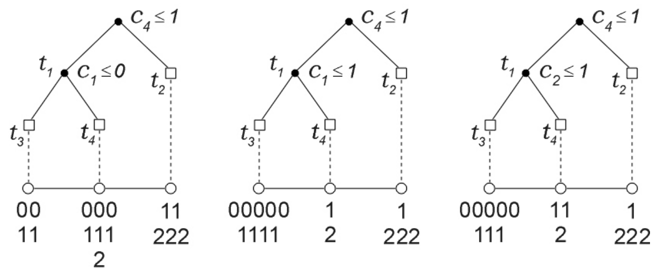
The next least pure node is t_3, and as can be seen in Figure 7.12, we find one best split, namely $c_2 \leq 1$.

Continuing like this, we finally find the tree presented in Figure 7.13. It can be seen that the influence of the inherent reversed preference between Sri Lanka and Thailand causes overfitting at node t_{14}. This can be seen intuitively by noting that we are better off not splitting t_{14} because it produces only counterintuitive (non-monotone) results.



splits of t_{root}	$(c_1,0)$	$(c_1,1)$	$(c_2,0)$	$(c_2,1)$	$(c_3,0)$	$(c_3,1)$	$(c_4,0)$	$(c_4,1)$
# solvable rev. pref.	6	2	3	6	1	7	4	1

Figure 7.10: Some of the possible splits of the root and the table of all splits with the corresponding number of solvable reversed preferences.



splits of t_1	$(c_1,0)$	$(c_1,1)$	$(c_2,0)$	$(c_2,1)$	$(c_3,0)$	$(c_3,1)$	$(c_4,0)$
# solvable rev. pref.	7	1	2	1	2	4	5

Figure 7.11: Some of the possible splits of t_1 and the table of all splits with the corresponding number of solvable reversed preferences.

splits of t_3	$(c_1,0)$	$(c_2,0)$	$(c_2,1)$	$(c_3,0)$	$(c_3,1)$	$(c_4,0)$
# solvable rev. pref.	7	2	1	2	4	3

Figure 7.12: Splits for t_3.

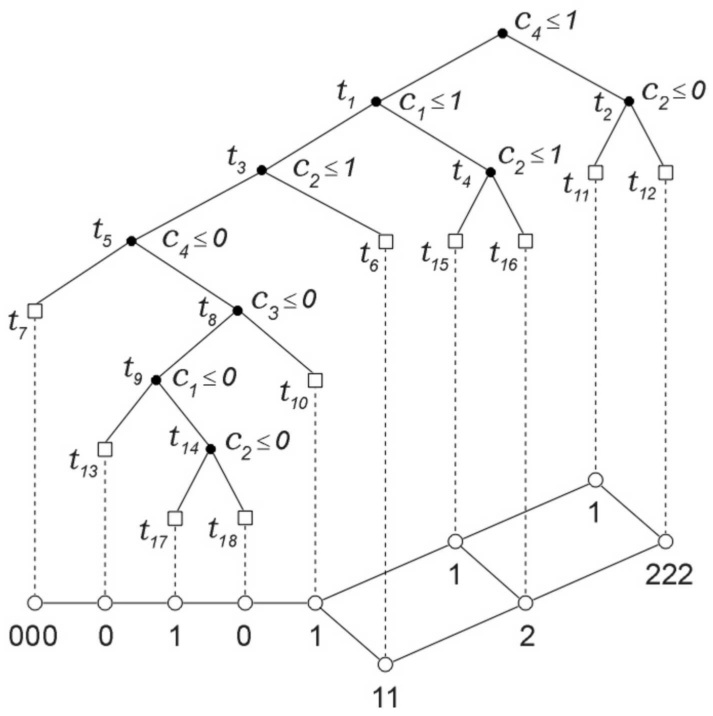


Figure 7.13: Overly grown tree for the contraception data.

Blind splits. As in the foregoing examples, we were lucky that none of the leaves ended up empty. Moreover, there is still one problem during the growing phase left untreated: what if we do not find a split that reduces either doubt or reversed preference? Consider for example Figure 7.14(a), and assume that for some reason

(a) Table.

	c_1	c_2	c_3	d
a_1	2	2	1	B
a_2	1	1	2	G

(b) Forced tree.

(c) Projection of the induced partitions for the two possible splits of t_1.

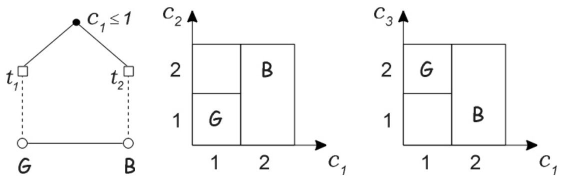


Figure 7.14: A possible problem during splitting: blind splits.

the first split was forced to be $c_1 \leq 1$. We continue splitting because our stopping demand has not yet been met, but how do we choose the split for t_1? There is only one example in t_1, and neither of the two possible splits $c_2 \leq 1$ and $c_3 \leq 1$ eliminates the existing reversed preference. Since both splits end up in a strict tie, the usual approach is to take the first one, that is $c_2 \leq 1$. In other words, the split is taken **blindly**. However, from Figure 7.14(c) it is clear we should have taken $c_3 \leq 1$ to be able to solve the reversed preference when splitting t_2. We will address this kind of problems[12] in Section 7.5. It should be noted that this problem is closely related to the problem of empty nodes.

BLIND SPLIT.

7.4 Principles of labelling and pruning

We already mentioned the need of pruning a tree, certainly in the ranking problem context. Obviously, pruning is inextricably intertwined with validation, such as the minimisation of some error functional R and the acceptability of the final tree to the user. On its turn, validation depends on how the leaves of the tree are labelled. You may have such a nice tree, but if your labelling rule tells you to assign always the same label to each leaf, you might as well prune the entire tree to its root.

7.4.1 Labelling

Introduction. In Chapter 4 we discussed circumstantially how to produce a monotone labelling scheme in the presence of reversed preference. In Chapter 5, we developed the OSDL algorithm (Ordinal Stochastic Dominance Learner) based on

[12]Remark that the "cornering" method described in [74, 89] (see also Section 3.2.3, p. 62) solves this particular example nicely, but as we discuss in Appendix 3.A, this is no longer true in more complex situations,where the problem of blind splits occurs frequently.

these ideas. Then, in Chapter 6, in particular Section 6.5.4 (see p. 174), we showed how everything could be adapted towards partition-based patterns.

The labelling rule.

IN GENERAL , a **labelling rule** $\lambda_T : \widetilde{T} \to \mathcal{L}_T$ assigns a label to each leaf of the tree. In order to avoid confusion, we stress that $\mathcal{L}_T$ may be different from the initial label set $\mathcal{L}$ from the learning sample. For example, we may have $\mathcal{L}_T = 2^{\mathcal{L}}$, the power set of $\mathcal{L}$, or $\mathcal{L}_T = \mathcal{F}(\mathcal{L})$, the set of distributions over $\mathcal{L}$.

LABELLING RULE
λ_T.
$\mathcal{L}_T$
$\mathcal{F}(\mathcal{L})$

FOR CLASSIFICATION trees, a labelling rule has only one objective: given the current tree, minimise the risk functional on some sample (this is usually the learning sample itself). Mostly, either $\mathcal{L}_T = \mathcal{F}(\mathcal{L})$ and the leaves are labelled with the distribution over $\mathcal{L}$ of the examples in each leaf, or $\mathcal{L}_T = \mathcal{L}$ where the label with the maximum probability in the previous distribution is assigned to the leaf.

FOR RANKING trees, a labelling rule has the additional requirement that it has to make the tree monotone. Possible candidates are the Minimal and Maximal extension mentioned in Section 5.4.2 (see p. 130) adapted towards partitions, but we opt for the modified OSDL algorithm, i.e.

$$\lambda_T(t) := \lambda_{\text{OSDL}}(t)\,,$$

because of its good performance[13] in ranking problems (see Section 5.4.5, p. 135).

About the monotonicity of trees. In [89], An *"efficient algorithm for testing the monotonicity of a tree"* was put forward. It is interesting to note that this algorithm follows as a consequence from Lemma 6.5.2 (see p. 170) and the fact that $t \preceq_T t' \iff a(t) \leq_{\mathcal{X}} b(t')$. Indeed, they imply that the partition-based monotonicity constraint can be rewritten as

$$a(t) \leq_{\mathcal{X}} b(t') \Rightarrow \lambda_T(t) \trianglelefteq_{\text{Im}} \lambda_T(t')\,.$$

Because Lemma 6.5.1 (see p. 170) says that partition-based monotonicity implies elementary monotonicity (i.e. on the pairs of vectors of $\mathcal{X}$), this means that a tree is monotone if it passes the algorithm

```
for all pairs of leaves $t, t' \in \widetilde{T}$ do
  if $\Big(a(t) \leq_{\mathcal{X}} b(t')$ and $\lambda_T(t) \ntrianglelefteq_{\text{Im}} \lambda_T(t')\Big)$ or
      $\Big(a(t') \leq_{\mathcal{X}} b(t)$ and $\lambda_T(t') \ntrianglelefteq_{\text{Im}} \lambda_T(t)\Big)$ then
    stop: $T$ not monotone
  end if
end for
```

which corresponds exactly to the algorithm described in [89] if $(\mathcal{L}_T, \trianglelefteq_{\text{Im}})$ is chosen to be $(\mathcal{L}, \leq_{\mathcal{L}})$.

[13] Of course, the effect of different labelling rules could (should) be compared experimentally.

7.4.2 Pruning

Introduction

IN CLASSIFICATION , it is observed that estimates based on small samples are *potentially* unreliable. They tend to capture patterns that are very specific to this particular learning sample. This is called overfitting. However, Shaffer [98] made clear that there are no *"statistical reasons for believing that these overfitting avoidance strategies do increase accuracy"*. This seems to be very true if a classification tree algorithm is let loose on a ranking problem: pruning the tree seems rather to decrease the accuracy than to increase it, as can be seen in the tables in Section 5.4.5 (see p. 135). This can be explained by observing that while the samples do get smaller, they also get more restricted by the monotonicity constraint. For example, in Figure 7.15 it is shown how only two examples $(a, d(a) = 2)$ and $(b, d(b) = 2)$ with

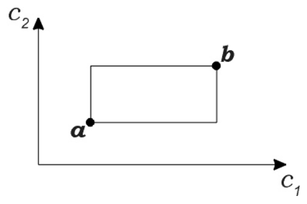


Figure 7.15: Two examples determine completely the labelling of a range of subsets of $\mathcal{X}$.

$\mathbf{a} \leq_{\mathcal{X}} \mathbf{b}$ fix the labels of all sets A within the interval $[\mathbf{a}, \mathbf{b}]$. Such a set A may even *not* contain any examples at all, while still leading to high accuracy.

HOWEVER, in the ranking tree approach, we also have to consider the problem of reversed preference, that urges us to consider pruning. And then there is this last, but ever so important reason: pruning *always* reduces the complexity of the tree. If you just want high accuracy, there are plenty of black box algorithms that will outperform trees. The main reason why trees are still popular is probably because they are so easy to interpret while still returning good accuracy, and a smaller tree is even easier to interpret.

Possible pruning techniques. Once the labelling rule has been fixed, we can apply any pruning approach we desire. But, as always, there is a catch. The same kind of woes we encountered during the growing phase emerge again, playing their dirty little tricks on the pruning process. Pruning algorithms are usually specifically created to be efficient for non-interacting leaves: for each node t, we can, given $\lambda_T(t)$, simply calculate the error $R(t)$ without having to worry that cutting some branches off will influence this result. For example, if we would interpret the tree of Figure 7.13 as a classification tree, then we can calculate the distribution-label of e.g. t_{11} as[14] $(0, 1, 0)$, which is nothing else but the distribution of the examples in

[14]If $\mathcal{L} = \{1, \dots, k\}$, then we denote a distribution $f_X \in \mathcal{F}(\mathcal{L})$ as a vector of dimension k: $f_X = (\mathcal{P}(X = 0), \dots, \mathcal{P}(X = k))$.

node t_{11}. Pruning node t_4 does not affect the number of examples in t_{11}. If we apply the labelling rule OSDL, then this sweet reality gets distorted: pruning t_4 does meddle with the distribution-label attached to t_{11}, instead of $(0, 1, 0)$, we would obtain $(0.5, 0.5, 0)$. As a consequence, although we can still keep the ideas of all pruning techniques, we can not rely on the implementations that are available for them. For that matter, we can not even easily extend these algorithms. For example, the efficient cost-complexity pruning algorithm described in [23, p. 293] heavily relies upon the inter-independence of the nodes. In fact the whole cross-validation technique for cost-complexity pruning is built upon the premisse that the leaves can be treated separately. Finding really efficient algorithms for pruning ranking trees is a difficult matter that will not be dealt with in this thesis.

Minimal cost-complexity pruning. Here we will briefly touch on the troubles that arise when we try to adopt the idea of cost-complexity pruning [23]. This approach creates a sequence $(T_0 = T(\alpha_0), \ldots, T_k = T(\alpha_k))$ of shorter and shorter nested trees, where each tree $T(\alpha_i)$ minimises the *cost-complexity measure*

$$R_\alpha(T) = R(T) + \alpha|\widetilde{T}|\,,$$

for ever increasing complexity parameter α_i (the cost per leaf is measured by the number of leaves). Here $R(T) = \sum_{t\in\widetilde{T}} R(t)$ is calculated on the training sample used to grow the tree. The second phase consists of finding reliable estimates for the $R(T_i)$. The final tree is then the one with the lowest estimated value of the risk functional.
For classification problems, $\lim_{\alpha\to\infty} \alpha$ corresponds to the shortest tree T_k (the root node). $\alpha = 0$ corresponds to the longest tree $T_0 = T_{\max}$, on the condition that the splitting rule is designed to keep reducing R at each split, and this is always the case in the existing classification tree algorithms. And here the problems start, for ranking trees, it is not guaranteed that R is non-increasing. This also implies that we cannot be sure the sequence $(T_i)_i$ consists of nested trees. This means that the whole theory of cost-complexity collapses.
Still, we can create a sequence of nested trees using an idea derived from cost-complexity pruning: pruning is done by repetitively cutting the weakest link in a tree T, we just can not be sure anymore that the trees in this sequence have the same nice properties as in the theory of cost-complexity pruning. The *weakest link* is defined as the inner node t that minimises the *link strength* [15]

$$\frac{R(T/t) - R(T)}{|\widetilde{T}_t| - 1}\,.$$

Remark that for classification problems, $R(T/t) - R(T) = R(t) - R(T_t)$, which saves a lot of calculations. For ranking problems, we need to establish for each possible pruned tree T/t the order $\leq_{T/t}$ and then apply OSDL to it. In Section 7.6, we will discuss how we can do this a bit more efficiently.

[15]The weakest link is the inner node t that produces the smallest value for α satisfying $R_\alpha(T/t) = R_\alpha(T)$. More details can be found in [23].

Remark. Independently from our research, monotone labelling and pruning techniques have been proposed in [17, 18, 87]. The post-pruning they offer, is based on the labelling of the leafs by the Minimal or Maximal extension as mentioned higher. The pruning is controlled by fixing a misclassification threshold percentage. Also a pre-pruning strategy is presented for the special case of the MDT (Monotone Decision Trees) algorithm [89] by another intervention in the updating rule Section 3.2.3 (see p. 62) of MDT. The growing of the is controlled by the minimal number of objects that have to fall into a leaf.

7.4.3 Example

In order to make this example easier to understand, we will use as a labelling rule the optimal (in terms of R chosen as the misclassification error w.r.t. to the given sample) monotone labelling. Remark that, for the small example considered here, it is easy to find such an optimal labelling, and that it is not necessarily unique. However, the optimal labelling problem becomes very complex in general, and has not yet been solved at the time of this writing. As a pruning rule, we will follow the weakest link strategy described above.
Consider again the tree obtained from Table 7.3, leading to the maximally grown tree shown in Figure 7.16 with $R(T_{\max}) = \frac{1}{2}$. At this point, we can prune any one

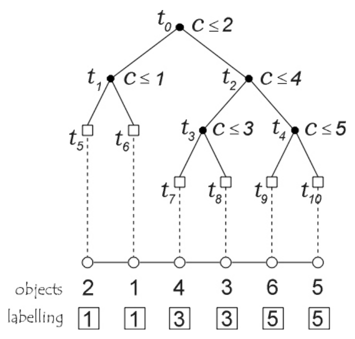
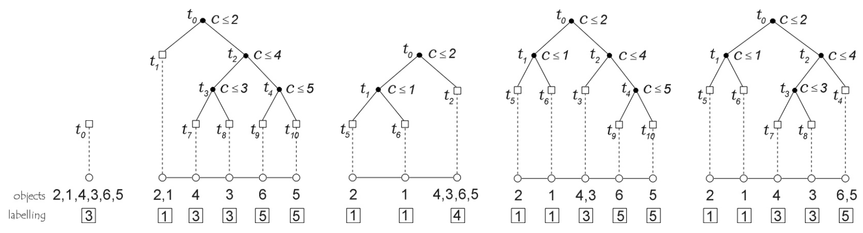


Figure 7.16: Maximally grown tree $T_{\max}$ on Table 7.3.

of the nodes t_0, t_1, t_2, t_3 or t_4, leading to the trees depicted in Figure 7.17. From the table in the same figure, we read that there are three weakest nodes t_1, t_2 or t_3. We choose the first one for pruning, resulting in the tree $T' = T/t_1$. We can now prune either t_0, t_1, t_3 or t_4. Continuing like this results in a nested sequence of trees. The pruning order and associated errors (on the training sample[16]) are shown in Table 7.5. From this table, we conclude that T_3 is the best tree, as depicted in Figure 7.18.

[16]In the optimal situation, this error should be estimated from a large independent sample set. Cross-validation techniques have not yet been developed for pruning ranking trees.



(a) Pruned tree T/t_0. (b) Pruned tree T/t_1. (c) Pruned tree T/t_2. (d) Pruned tree T/t_3. (e) Pruned tree T/t_4.

node	t_0	t_1	t_2	t_3	t_4
error pruned tree	$\frac{5}{6}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{1}{2}$	$\frac{1}{2}$
link strength	$\frac{1}{15}$	0	$\frac{1}{18}$	0	0

Figure 7.17: Pruning the tree grown on Table 7.3: pruning $T_{\max}$.

pruning sequence	T_0	T_1	T_2	T_3	T_4	T_5
	$T_{\max}$	$T_{\max}/t_1$	T_1/t_3	T_2/t_4	T_3/t_2	T_4/t_0
error	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{2}{3}$	1

Table 7.5: Pruning sequence for the tree grown on Table 7.3.

7.5 A ranking tree algorithm

Notions and conventions. Let R denote a binary relation on a set X. A **chain** in R is a sequence $(a_i)_i$ of elements $a_i \in X$ such that $a_i R a_{i+1}$.

CHAIN IN A BINARY RELATION.

A **single split** s is a split on one leaf t. We will denote the children of t after a single binary split by t_L and t_R with $t_L \le_T t_R$.

SINGLE SPLIT. t_L, t_R

Performing several single splits s_i, $i = 1, \ldots, k$ simultaneously, is called a **multiple split**. In the special case where such a multiple split s consists of several single binary splits of the form $s_i = (t_i, c, v)$, $i = 1, \ldots, k$ with $t_i \in \widetilde{T}$, we write this short as $s = (L, c, v)$, with $L = \{t_1, \ldots, t_k\}$. We stick to the notation $T(s)$ for denoting the tree resulting from T after performing the split s.

MULTIPLE SPLIT.

A single split is called **void** if it does not affect the partitioning of $\mathcal{X}$. For example, consider the tree in Figure 7.18 (see p. 200). The split $(t_1, c, 1)$ is a void split. More formally, if $t_{\mathcal{X}} = [\mathbf{x}, \mathbf{y}]$, then any split (t, c_i, v_i) with $v_i \notin [x_i, y_i[$ is a void split. Do remark that nothing prevents us from performing a void split, but the only result is that nothing will happen. However, it can prove helpful to consider the children t_L and t_R of such a split, where one of them equals t, and the other one is a **phantom node**, a non-existing node.

VOID SPLIT.

PHANTOM NODE.

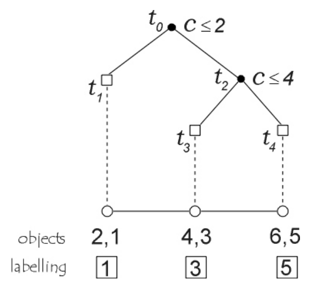


Figure 7.18: Final tree.

7.5.1 Avoiding blind splits

Introduction. Recall the problem we raised in Figure 7.14, where it was difficult to find a split for one of the leaves. The essence of this issue lies in the interdependency of the leaves. So instead of abiding by the idea of splitting a single leaf at a time, we should consider the option of splitting several leaves at a time.

Choosing which leaves to split. We let us inspire by the guideline of trying to minimise solvable reversed preferences. The idea is to split the couple of leaves that have the most solvable reversed preferences between them. If there are no reversed preferences, we only split one node, the impurest one. The algorithm **PickNodes** is described in Algorithm 7.1. We denote the number of solvable reversed preferences between two leaves $t, t' \in \widetilde{T}$ by $\#\text{rp}(t, t')$.

Algorithm 7.1 : PickNodes, primal selection of leaves to split

```
RP ← $\max_{(u,v)\in\widetilde{T}^2} \#\text{rp}(u, v)$;
if RP = 0 then
    return impurest leaf;
else
    return $(t, t') \leftarrow \arg\max_{(u,v)\in\widetilde{T}^2} \#\text{rp}(u, v)$;
end if
```

However, even though considering a simultaneous split of two leaves may solve the problem in Figure 7.14, blind splits (in the sense they can not foresee how to eliminate the reversed preference) may still occur. Consider for example the situation in Table 7.19. To keep the example simple, we use n-ary splits (like in ID3 [92]). If the first split is based on c_1, then we may choose whatever splits we like on t_1 and t_3, but we will never break the relation $a \triangleleft_p b$ because of the existence of t_2: any child t'_1 of t_1, and any child t'_3 of t_3 will always be related as $t'_1 \leq_T t_2 \leq_T t'_3$. Our personal observations have made clear this kind of situation is not exceptional, even in binary splitting.

(a) Part of the table.

	c_1	c_i	d
a	3	...	G
b	1	...	B

(b) Dominance graph of any split of (t_1, t_3).

(c) Induced partition.

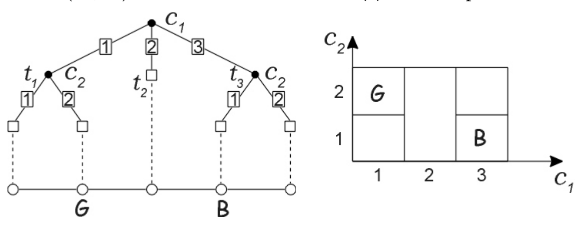


Figure 7.19: A constant dominance graph for splitting (t_1, t_3).

Avoiding blind splits. The problem of blind splits can be solved by not simply splitting the two leaves leading to reversed preference, but also all leaves that lie in between them.

Lemma 7.5.1. *Let $t, t' \in \widetilde{T}$, with $t <_T t'$, be two leaves such that there exists at least one solvable reversed preference between them. Denote $L = \{u \in \widetilde{T} \mid t \leq_T u \leq_T t'\}$. There exists a multiple split $s = (L, c, v)$ such that at least one of the solvable reversed preferences in T does not occur anymore in $T(s)$.*

Proof.
Let $C = \{c_i \mid i \in N\}$ be the set of criteria. By assumption, there exist some objects $a \in t$ and $b \in t'$, with $\mathbf{a} \not\leq_{\mathcal{X}} \mathbf{b}$ and $d(a) >_{\mathcal{L}} d(b)$, i.e. a and b lead to reversed preference in T because $t <_T t'$, but not within the learning sample. This means there exists at least one $i \in N$ such that $c_i(a) >_{c_i} c_i(b)$. We will now demonstrate that the splits (L, c_i, v_i) with $v_i \in [c_i(b), c_i(a)[$ will solve the reversed preference between a and b in $T(s)$. Let $s = (L, c, v)$ be such a split.

Consider any chain $(l_i)_{i=1}^k$ in $(\widetilde{T}, \leq_T)$ with $l_1 = t$ and $l_k = t'$. By definition of s, all these leaves are split following $c \leq v$. We have $a \in t_R = (l_1)_R$ and $b \in t'_L = (l_k)_L$. Moreover, it holds that $t_R \not\leq_{T(s)} t'_L$, and they are not connected by the children of the leaves l_i. Indeed, after splitting, the chain $(l_i)_i$ is split into two chains $((l_1)_L, (l_2)_L, \ldots, (l_k)_L, (l_k)_R)$ and $((l_1)_L, (l_1)_R, (l_2)_R, \ldots, (l_k)_R)$, as can be seen in Figure 7.20. Remark that some of these children may in fact be phantom nodes, but this does not affect our demonstration (see below). Since this

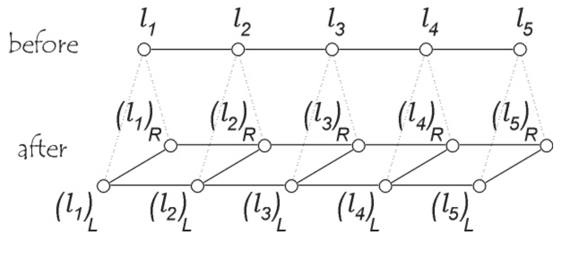
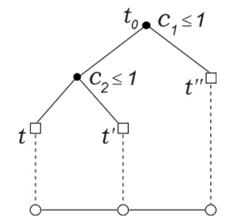
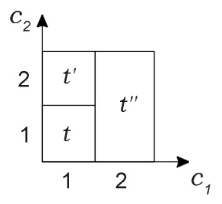


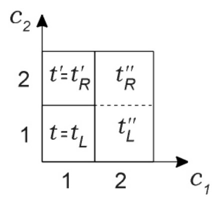
Figure 7.20: The splitting of the chain $(l_i)_i$.



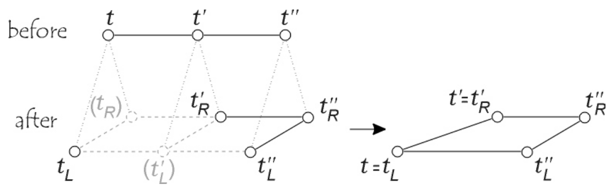
(a) Tree before splitting.



(b) Associated partition before splitting.



(c) Associated partition after splitting.



(d) Dominance graph with phantom nodes.

Figure 7.21: Phantom node.

holds for all chains connecting t and t', we ultimately have that $t_R \not\leq_T t'_L$, and therefore the reversed preference between a and b is resolved in $T(s)$. $\square$

Example of multiple split with phantom nodes. Consider the tree depicted in Figure 7.21(a), and the (multiple) split $c_2 \leq 1$ on the chain of nodes t, t', t''. Clearly, the splits on t and t' are void ones, with phantom nodes t_R and t'_L. As can be seen from Figure 7.21(d), considering the would-be relations with the phantom nodes does not affect the relations between the real nodes.

7.5.2 The splitting and pruning algorithms

Splitting. Once the initial node(s) to be split are chosen, we have to find the split to perform. The best split among a set of splits is always chosen in two steps: first find the one that minimises the number of solvable reversed preferences in $T(s)$, and amid these, choose the ones that minimise the twoing criterion on $T(s)$. If there are several such splits, pick the first one encountered.
Algorithm 7.2 describes a possible splitting algorithm for ranking trees. First choose the nodes to split based on **Picknode**, which delivers either one (case (i)) or two nodes (case (ii)). In the latter case, we know by Lemma 7.5.1 that we can always reduce the number of solvable reversed preferences by splitting all chains between the chosen nodes t_1 and t_2 (case (ii–c)). However, we first try to lower this number by only splitting t_1 or t_2 (case (ii–a)), or by a simultaneous split of t_1 and t_2 (case (ii–b)).
We denote the number of solvable reversed preference inside a tree T by $\#\mathrm{rp}(T)$.

Algorithm 7.2 : Split, find a split for a tree T

```
$L \leftarrow$ Picknode($T$);

(i)  case $L = \{t\}$:
     for all $i \in N$ do
         find the best split $s_i = (t, c_i, v)$ among the non-void splits of $t$;
     return $s \leftarrow$ best one among $s_i$.

(ii) case $L = \{t_1, t_2\}$:
     Rename such that $t_1 \leq_T t_2$. We proceed in three steps where we try to find
     a solution that requires the minimal number of nodes to be split:

     (a) for $i = 1, 2$ do
             $S_i \leftarrow \{(t_i, c, v) \mid s = (t, c, v)$ in $T,\ t \in$ ancestors$(t_i)\}$,
             $s_i \leftarrow$ best split in $S_i$;
         if ($s \leftarrow$ best split in $\{s_1, s_2\}$) is such that $\#\text{rp}(T(s)) < \#\text{rp}(T)$ then
             return $s$;

     (b) for all $i \in N$ do
             find the best split $s_i = (\{t_1, t_2\}, c_i, v)$ among those splits for
             which $(t_1, c_i, v)$ and $(t_2, c_i, v)$ are both non-void;
         if ($s \leftarrow$ best among $s_i$) is such that $\#\text{rp}(T(s)) < \#\text{rp}(T)$ then
             return $s$;

     (c) $L \leftarrow \{t \in \widetilde{T} \mid t_1 \leq_T t \leq_T t_2\}$;
         for all $i \in N$ do
             $S_i \leftarrow \{(L, c_i, v) \mid v \in [a(t_2)_i, b(t_1)_i[\ \}$,
             $s_i \leftarrow$ best split in $S_i$;
         return $s \leftarrow$ best one among $s_i$.
```

As a variant on step (ii–b), we might consider to search s_i among the set of splits $\langle (t_1, c_i, v), (t_2, c_i, w) \rangle$ where $v \geq_c w$ and both simultaneous splits are non-void. In case $t_1 \prec_T t_2$, such splits still ensure that $(t_1)_R \not\leq_T (t_2)_L$.

Pruning. We implemented the "cut-the-weakest-link" approach based on the minimal cost-complexity pruning technique, as described in Section 7.4.2, to obtain a sequence of smaller and smaller nested trees $(T_0 = T_{\max}, \ldots, T_k = t_{\text{root}})$. What we did not yet address, is the problem of picking out the final tree from the obtained pruning sequence.

For classification trees, there are two approaches, either use an additional independent (large enough) learning sample for pruning, or cross-validation. At this moment, there is not yet a theory allowing cross-validation for ranking trees, so we are left with the first option. However, if no such sample is available, the final tree can always be chosen from the sequence using the original learning sample. As we explained above, our splitting does not guarantee the continuous decreasing of the risk functional R on the training set: we even observed that, for non-monotone data sets, the plot of R in function of the number of leaves shows a curve starting high, then going down steeply, reaching a minimum and climbing up again. This can be explained by noticing that at a certain point, the decrease in the number of solvable reversed preferences is no longer sufficient to compensate for the growing number of unsolvable reversed preferences.

7.5.3 Experiments

Introduction In this section, we present the results of some experiments we conducted. The general settings and implementation frame are the same as described in Sections 5.4.1 to 5.4.3 (see p. 130). We only conducted experiments using artificially generated monotone data sets, as described in the next paragraph.

Artificially generated data sets. Our experiments for RT, Ranking Trees, are based on artificially generated monotone samples $\mathcal{S}$, where we divide $\mathcal{S}$ into a training set and a test set. The size of the test set will always be 500. All results gathered in the figures and tables are averages from 10 independent runs.

We consider 3 monotone designs, and 1 non-monotone design (a projection of a monotone data set as discussed in Section 5.4.1 (see p. 130)) whose characteristics can be found in Table 7.6.

	design 1	design 2	design 3	design 4
$\|C\|$ (# generated criteria)	$*$	7	6	8
$\|C'\|$ (# used criteria)	$*$	7	6	6
$\|\mathcal{X}_c\|$ (# criterion values)	5	$*$	5	5
$\|\mathcal{L}\|$ (# labels)	4	4	4	4
$\|\mathcal{S}\|$ (# learning instances)	100	100	$*$	$*$

Table 7.6: Characteristics of the different designs.

COMPARISON WITH TREE ALGORITHMS. We first perform some comparisons with C4.5 [93], both pruned and unpruned variants, and MDT, Monotone Decision Trees [89]. We calculated the classification accuracy, Kendall's tau, the mean absolute error (MAE) and the mean square error (MSE). More details about these measures can be found in Section 5.4.3 (see p. 132). The results for designs 1 and 2 are shown in Figures 7.22 and 7.23.

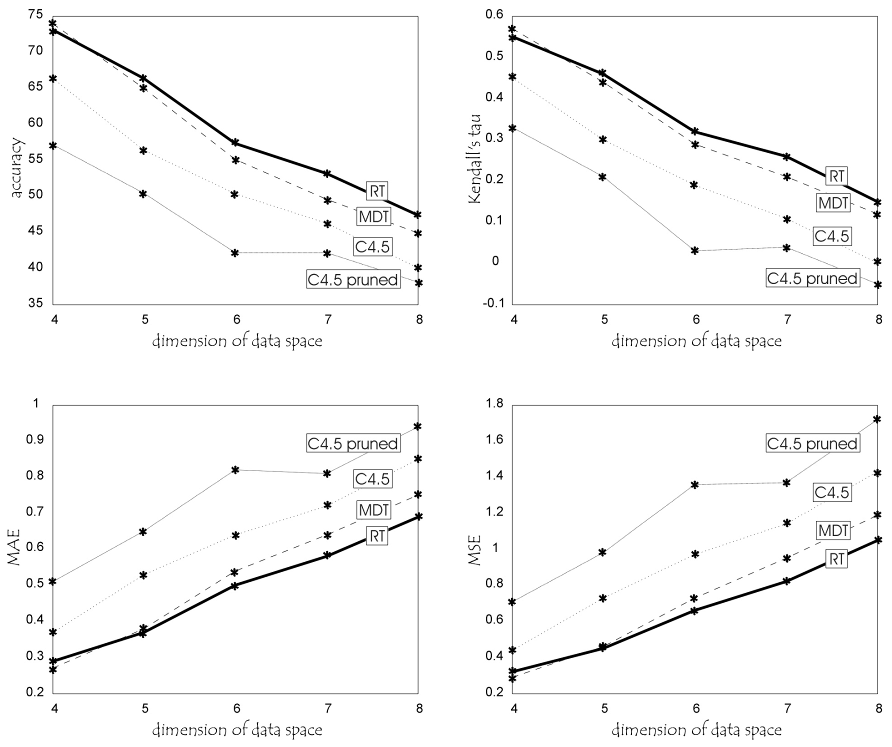


Figure 7.22: Design 1, variable dimension.

The first noticeable feature of these graphs is the similar behaviour in these experiments of the classification accuracy and Kendall's tau on the one hand, and of the MAE and MSE on the other. Therefore, for the remainder of this section, we will just concentrate on the classification accuracy and the MAE.
Secondly, these figures indicate that RT produces trees with slightly better results compared to MDT. To get an idea of the significance of this improvement, we performed the one-sided Wilcoxon signed-rank test [68], which is non-parametric. The results are given in Table 7.7. From these, it becomes clear that the larger the space, the more important and significant the gain becomes. This gain can be attributed to the avoidance of blind splits, as is evidenced from the average tree sizes (number of leaves) for these two experiments as summarised in Table 7.8.

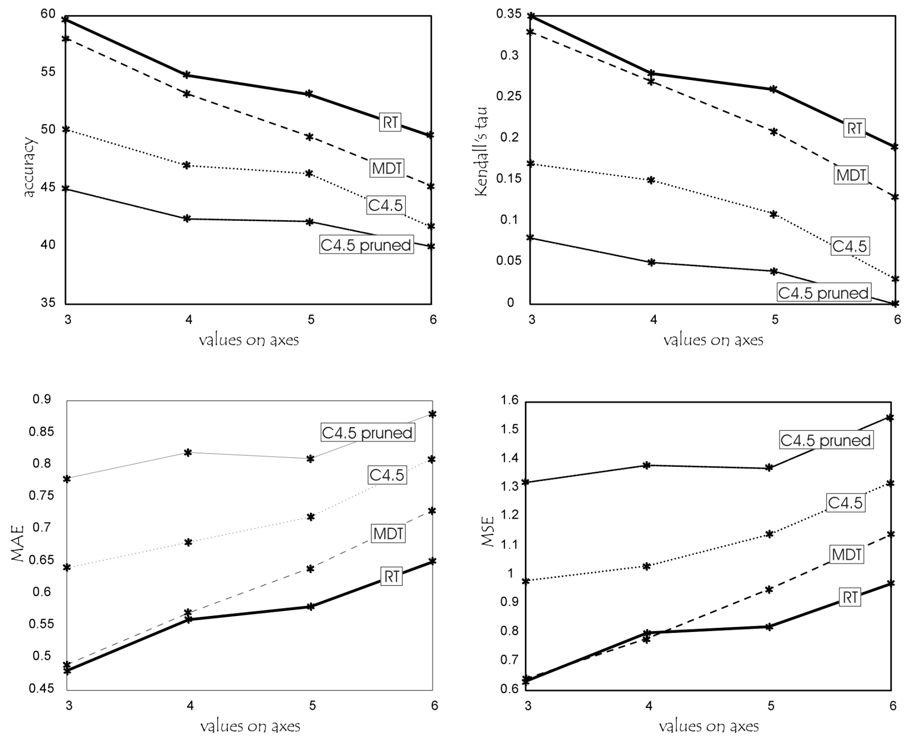


Figure 7.23: Design 2, variable axis length.

COMPARISON WITH OTHER ALGORITHMS. Now, we also contrast the tree algorithms with OSDL, the Ordinal Stochastic Dominance Learner from Section 5 (see p. 115), and the Naive Bayes method. The results on the monotone design 3 and the non-montone design 4 (both with variable training sample size) are shown in Figures 7.24 and 7.25. It seems that on design 3, RT still improves on OSDL w.r.t. the classification accuracy. There is a price to be paid however, namely a slight deterioration on the MAE. For the non-monotone design 4, RT tastes defeat from OSDL, except on accuracy when there is really little data available.

7.6 More efficient splitting and pruning

a, a', a_1, a_2

Notions and conventions. To keep notations manageable, we denote $a = a(t)$, $a' = a(t'), a_1 = a(t_1), a_2 = a(t_2)$ and similarly for b. With the notation $a(t)_i$, we refer to the i^{th} component of the vector $a(t)$. Because this section is only concerned with $\mathcal{X}$, this cannot lead to confusion with objects from Ω.

(a) Design 1 (monotone), variable dimension.

	mean accuracy			mean MAE		
dim	alt. hypothesis	Z-value	p-value	alt. hypothesis	Z-value	p-value
4	RT $<$ MDT	-0.4587	0.3232	RT $>$ MDT	1.0213	0.1536
5	RT $>$ MDT	1.5799	0.0571	RT $<$ MDT	-1.173	0.1204
6	RT $>$ MDT	2.0412	0.0206	RT $<$ MDT	-2.2949	0.0109
7	RT $>$ MDT	1.8371	0.0331	RT $<$ MDT	-2.0909	0.0183
8	RT $>$ MDT	2.7557	0.0029	RT $<$ MDT	-2.3474	0.0095

(b) Design 2 (monotone), variable axis length.

	mean accuracy			mean MAE		
$\lvert\mathcal{X}_c\rvert$	alt. hypothesis	Z-value	p-value	alt. hypothesis	Z-value	p-value
3	RT $>$ MDT	1.478	0.0697	RT $<$ MDT	-0.7645	0.2223
4	RT $>$ MDT	0.867	0.193	RT $<$ MDT	-0.102	0.4594
5	RT $>$ MDT	1.8371	0.0331	RT $<$ MDT	-2.0909	0.0183
6	RT $>$ MDT	2.5499	0.0054	RT $<$ MDT	-2.2934	0.0109

Table 7.7: One-sided Wilcoxon signed-rank test for designs 1 and 2.

(a) Design 1, variable dimension.

dim	C4.5pruned	C4.5	MDT	RT
4	13.2±3.55	24.2±4.61	56.9±12.72	47.3±6.25
5	12.4±1.78	27.6±2.76	171.0±41.33	73.3±11.78
6	9.7±3.86	29.1±2.88	706.7±305.6	99.3±20.58
7	11.0±3.02	32.8±4.71	3340.5±1739.84	130.7±47.0
8	9.2±2.49	32.7±4.4	10044.1±4116.59	173.1±69.77

(b) Design 2, variable axis length.

$\lvert\mathcal{X}_c\rvert$	C4.5pruned	C4.5	MDT	RT
3	11.2±2.86	30.9±4.89	224.4±87.39	94.5±23.28
4	9.3±2.21	32.3±3.43	816.7±224.83	113.1±22.03
5	11.0±3.02	32.8±4.71	3340.5±1739.84	130.7±47.0
6	10.3±2.83	33.8±4.13	7846.5±3240.36	160.1±62.11

Table 7.8: Number of leaves.

7.6.1 Introduction

As we have seen, the order of the leaves is of primordial importance in the construction of ranking trees. Both during the growing (splitting) phase, as in the pruning phase.

Splitting. Let T be a not fully grown tree. To choose the next split, we must evaluate the splits s from some set S of possible splits according to some splitting criterion, e.g. minimising the number of reversed preferences in the split tree $T(s)$. Hence, for each split (single or multiple) $s \in S$, the partial dominance order $\leq_{T(s)}$ of the leaves $\widetilde{T(s)}$ must be ascertained. If this has to be done for each split s on basis of the definition of the partial dominance order, this would mean the pairwise comparison of all leaves to obtain the partial dominance relation $\preceq_{T(s)}$, followed

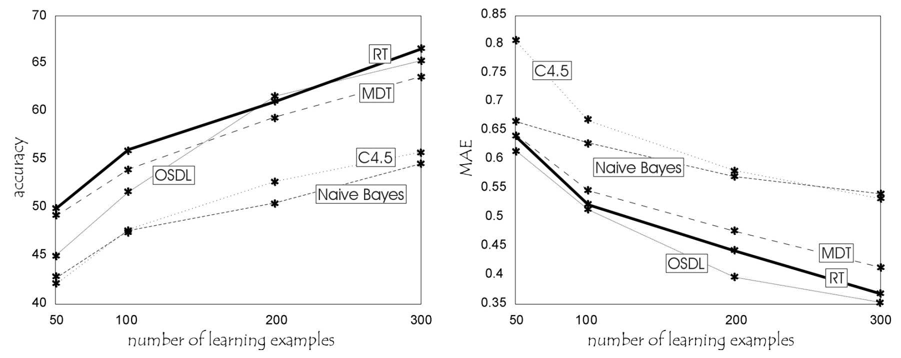


Figure 7.24: Design 3 (monotone), variable training sample size.

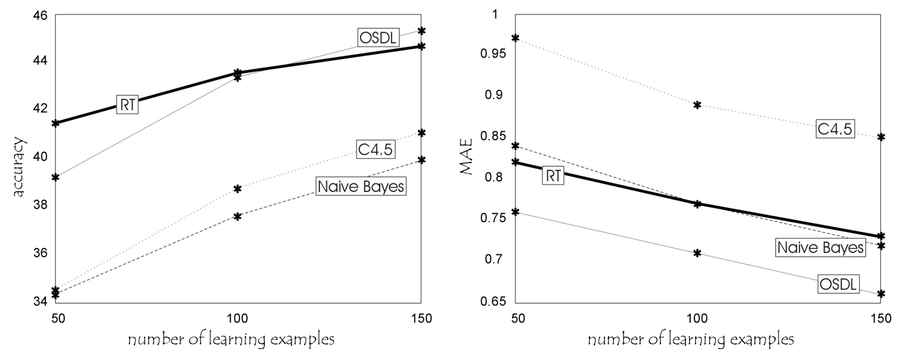


Figure 7.25: Design 4 (non-monotone), variable training sample size.

by its transitive closure to obtain $\leq_{T(s)}$. This is clearly a computationally intensive process. While it is fairly obvious how to obtain $\preceq_{T(s)}$ from $\preceq_T$ with a minimum of additional calculations, this still leaves unaddressed the transitive closure, which is the bigger time consumer of the two. Theorem 7.6.9 will provide an answer to how $\leq_{T(s)}$ can be computed from $\preceq_T$ and $\leq_T$ and a minimum of additional calculations.

Pruning. A similar problem occurs during the pruning process. In order to obtain the next pruned tree T_{i+1} from T_i in the pruning sequence, we must compare all trees T'_{i+1} that can be obtained from T_i via pruning. Consider for example weakest-link pruning: since there are $2|\widetilde{T}_i| - 1$ nodes (including the leaves) in the binary tree T_i, there are $|\tilde{T}_i| - 1$ such trees T'_{i+1}. For each of these trees, the nodes must be labelled monotonically, in our case by the OSDL algorithm. Just running OSDL on each of the trees T'_{i+1} requires huge computation times, mainly because each call to OSDL implies that the order $\leq_{T'_{i+1}}$ must be calculated. As before, this is in essence a computation intensive process. We show in Theorem 7.6.8 how this can be done more efficiently by deriving $\leq_{T'_{i+1}}$ directly from $\leq_{T_i}$.

7.6.2 Some lemmas

Splitting is inherently more complex than pruning. This also surfaces during the following demonstrations: we will need pruning properties to capture the behaviour of splitting.

Preliminaries. If we split a leaf t by some univariate binary split (t, c_i, v) into the children t_1 and t_2, with $t_1 \preceq_T t_2$, then we have that

$$a_1 = a, \quad b_2 = b, \quad a_1 <_{\mathcal{X}} a_2, \quad b_1 <_{\mathcal{X}} b_2, \quad (b_1)_i = v, \quad (a_2)_i = v\,.$$

This is exemplified in Figure 7.26.

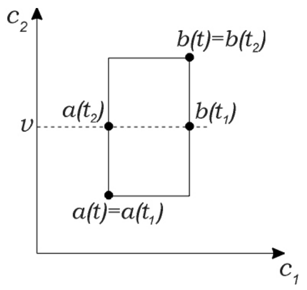


Figure 7.26: Split t into $t_1 \prec_T t_2$ using the split $c_2 \leq v$. $a(t_1) \leq_{\mathcal{X}} a(t_2)$.

About pruning and about cycles.

Lemma 7.6.1. *Consider a univariate binary tree T and a leaf $t \in \widetilde{T}$. Let T' denote the tree we obtain from T by the univariate splitting of the node t into the nodes t_1 and t_2, with $t_1 <_{T'} t_2$. For any $t' \in \widetilde{T'} \setminus \{t_1, t_2\}$ it holds that*

$$t_1 \prec_{T'} t' \Rightarrow t \prec_T t' \qquad \textit{and} \qquad t_2 \prec_{T'} t' \Rightarrow t \prec_T t'\,.$$

Likewise

$$t' \prec_{T'} t_1 \Rightarrow t' \prec_T t \qquad \textit{and} \qquad t' \prec_{T'} t_2 \Rightarrow t' \prec_T t\,.$$

Proof.
If $t_1 \prec_{T'} t'$, then $a = a_1 <_{\mathcal{X}} b'$ and therefore immediately $t \prec_T t'$. If $t_2 \prec_{T'} t'$, then $a = a_1 <_{\mathcal{X}} a_2 <_{\mathcal{X}} b'$ and therefore again $t \prec_T t'$. The other directions are similar. □

Proposition 7.6.2. *Let T be a univariate binary tree. The relation $\prec_T$ contains no cycles.*

Proof.
The proof is done by induction: if T_0 is a tree with one node, the root, then obviously $<_{T_0}$ does not contain any cycles. The same is true for the tree T_1 after the first split: if $t_1 \prec_{T_1} t_2$ and $t_2 \prec_{T_1} t_1$, then $a_1 <_{\mathcal{X}} b_2$ and $a_2 <_{\mathcal{X}} b_1$. But this would imply by Lemma 7.2.2 that $t_1 \cap t_2 \neq \emptyset$, a contradiction.
Now assume that the tree T_i contains no cycles, and let T_{i+1} be the tree derived from T_i by the univariate binary splitting of the leaf $t \in \widetilde{T}_i$. Assume T_{i+1} contains a cycle $t_1 \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_k \prec_{T_{i+1}} t_1$. Because $\prec_{T_i}$ does not contain any cycles, at least one of the t_i must be a child of t. Moreover, if the same child occurs at least twice in the cycle, we can always consider a smaller sub-cycle containing each child at most once. We consider two cases:

(i) Only one of the t_i is a child of t:

$$t_1 \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_{j-1} \prec_{T_{i+1}} t_j = t_{\text{child}} \prec_{T_{i+1}} t_{j+1} \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_k \prec_{T_{i+1}} t_1 \,.$$

Now Lemma 7.6.1 implies that

$$t_{j-1} \prec_{T_i} t \prec_{T_i} t_{j+1} \,,$$

which means $\prec_{T_i}$ contains a cycle. A contradiction.

(ii) Both children t_L and t_R are contained exactly once in the cycle, $t_j = t_L$ and $t_k = t_R$.

(a) We suppose that $j < k$. If $k \neq j+1$, we may construct another cycle in $\prec_{T'}$ such that t_L and t_R are subsequent (we know that $t_L \prec_{T'} t_R$):

$$t_1 \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_{j-1} \prec_{T_{i+1}} t_j = t_L \prec_{T_{i+1}} t_k = t_R \prec_{T_{i+1}} t_{k+1} \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_k \prec_{T_{i+1}} t_1 \,.$$

We find again that

$$t_{j-1} \prec_{T_i} t \prec_{T_i} t_{j+2} \,,$$

implying the existence of a cycle in $\prec_{T_i}$.

(b) We suppose that $k < j$. The configuration $j = k+1$ is excluded by Lemma 7.2.2. So, we have as a subsequence:

$$t_k = t_R \prec_{T_{i+1}} t_{k+1} \prec_{T_{i+1}} \ldots \prec_{T_{i+1}} t_{j-1} \prec_{T_{i+1}} t_j = t_R \,,$$

implying

$$t \prec_{T_i} t_{k+1} \prec_{T_i} \ldots \prec_{T_i} t_{j-1} \prec_{T_i} t \,,$$

which is a cycle in $\prec_{T_i}$. □

Corollary 7.6.3. *If T is a univariate binary tree, then the relation $<_T$ contains no cycles.*

Proof.
The transitive closure cannot introduce cycles. □

We can now generalise Lemma 7.6.1 towards the order $\leq_T$ on $\widetilde{T}$.

Lemma 7.6.4. *Consider a univariate binary tree T and a leaf $t \in \widetilde{T}$. Let T' denote the tree we obtain from T by the univariate splitting of the node t into the nodes t_1 and t_2, with $t_1 <_{T'} t_2$. For any $t' \in \widetilde{T'} \setminus \{t_1, t_2\}$ it holds that*

$$t_1 <_{T'} t' \Rightarrow t <_T t' \qquad \textit{and} \qquad t_2 <_{T'} t' \Rightarrow t <_T t' \,.$$

Likewise

$$t' <_{T'} t_1 \Rightarrow t' <_T t \qquad \textit{and} \qquad t' <_{T'} t_2 \Rightarrow t' <_T t \,.$$

Proof.

(i) Assume $t_2 <_{T'} t'$, but $t_2 \not\prec_{T'} t'$. In that case, there exist leaves $l_i \in \widetilde{T'} \setminus \{t_1\}$ such that $t_2 \prec_{T'} l_1 \prec_{T'} \ldots \prec_{T'} l_k \prec_{T'} t'$. Lemma 7.6.1 learns us that $t \prec_T l_1$. From Proposition 7.6.2 we already know that the chain $(l_i)_i$ will not contain t_1 or t_2. This means that the chain $(l_i)_i$ also exists in $\prec_T$. So in all, transitivity leads to $t <_T t'$.

(ii) Assume $t_1 <_{T'} t'$, but $t_1 \not\prec_{T'} t'$. We consider two cases: (a) one of the $l_i = t_2$, (b) none of the $l_i = t_2$. Assume (a) holds, from the previous, we immediately have $t <_T t'$. Now assume that (b) holds. As above, we find from Lemma 7.6.1 and Proposition 7.6.2 that $t <_T t'$.

(iii) The other direction is similar. □

The previous lemma is visualised in Figure 7.27.

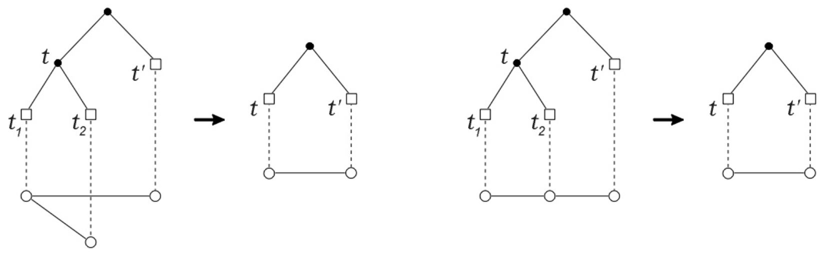


Figure 7.27: Visual presentation of Lemma 7.6.4.

Lemma 7.6.5. *Consider a univariate binary tree T and a leaf $t \in \widetilde{T}$. Let T' denote the univariate binary tree we obtain from T by splitting the node t into the nodes t_1 and t_2, with $t_1 <_{T'} t_2$. There exists no leaf t' from $\widetilde{T} \setminus \{t\}$ such that $t_1 <_{T'} t' <_{T'} t_2$.*

Proof.
Assume that such a leaf exists, then Lemma 7.6.4 tells us that $t <_T t' <_T t$, which is impossible because of Corollary 7.6.3. □

About splitting.

Lemma 7.6.6. *Consider a univariate binary tree T and a leaf $t \in \widetilde{T}$. Let $s = (t, c_i, v)$ be a single univariate binary split on T and denote $T(s)$ the tree we obtain from T by splitting the node t into the nodes t_1 and t_2, with $t_1 <_{T(s)} t_2$, using the split s. For any $t' \in \widetilde{T(s)} \setminus \{t_1, t_2\}$ (or equivalently, $t' \in \widetilde{T} \setminus \{t\}$), it holds that:*

(i) *If $t' <_T t$, then either $t' <_{T(s)} t_1$ or $t' \,\|_{T(s)}\, t_1$, and always $t' <_{T(s)} t_2$.*

(ii) *If $t <_T t'$, then always $t_1 <_{T(s)} t'$, and either $t_2 <_{T(s)} t'$ or $t_2 \,\|_{T(s)}\, t'$.*

(iii) *If $t \,\|_T\, t'$, then both $t_1 \,\|_{T(s)}\, t'$ and $t_2 \,\|_{T(s)}\, t'$.*

The same holds for $\prec_T$.

Proof.

(i) Assume that $t' <_T t$. If $t_1 <_{T(s)} t'$ then Lemma 7.6.4 results in $t <_T t'$, a contradiction. So we have $t_1 \not<_{T(s)} t'$. Concerning the relation between t' and t_2, we distinguish between two cases:

- Case 1: $a' <_{\mathcal{X}} b$. Since $b = b_2$, we have $t' <_{T(s)} t_2$.
- Case 2: there exist leaves $l_i \in \widetilde{T} \setminus \{t\}$ such that $t' <_T l_1 <_T \ldots <_T l_k <_T t$ with $a' <_{\mathcal{X}} b(l_1)$, $a(l_i) <_{\mathcal{X}} b(l_{i+1})$ and $a(l_k) <_{\mathcal{X}} b$. We know from the previous that $l_k <_{T(s)} t_2$, and therefore by transitivity that $t' <_{T(s)} t_2$.

(ii) Similar to (i).

(iii) Assume that $t \,\|_T\, t'$. If we would have $t_1 <_{T(s)} t'$, then Lemma 7.6.4 implies that also $t <_T t'$, a contradiction. The relations $t' <_{T(s)} t_1$, $t_2 <_{T(s)} t'$ and $t' <_{T(s)} t_2$ also lead to a contradiction. □

This lemma can be visualised as in Figure 7.28.

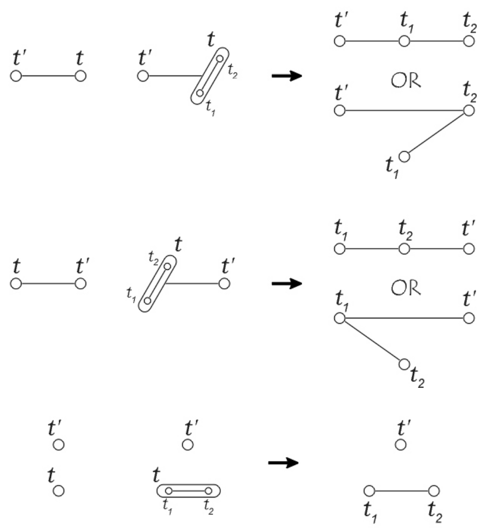


Figure 7.28: Visual presentation of Lemma 7.6.6.

7.6.3 Main theorems

The previous lemmas are very important for more efficient splitting and pruning of the tree. They provide a way of rebuilding the new relations $<_{T'}$ from $<_T$, whether T' is obtained from T because a leaf of T was split, or an internal node of T was pruned.

Pruning. The next lemma considers the case where a branch T_t of depth 1 is pruned.

Lemma 7.6.7. *Consider a univariate binary tree T and an inner node t of T whose children t_1, t_2 are leaves of T. Let T/t denote the tree we obtain from T by pruning the node t. For all $u, u' \in \widetilde{T/t} \setminus \{t\}$ it holds that*

$$u <_T u' \Rightarrow u <_{T/t} u' .$$

Proof.
Assume $u <_T u'$. If $a(u) <_{\mathcal{X}} b(u')$, then immediately $u <_{T/t} u'$. Otherwise, there exists at least one chain of leaves $u = l_1 \prec_T \ldots \prec_T l_k = u'$. If one of these chains $(l_i)_i$ consists only of leaves from $\widetilde{T} \setminus \{t_1, t_2\}$ then this chain exists also in T/t, whence $u <_{T/t} u'$. Now assume that each of these chains $(l_i)_i$ contains t_1 and/or t_2. Remark that because $<_T$ does not contain any cycles, the nodes t_1 and t_2 can appear at most once in these chains $(l_i)_i$. Let $(l_i)_i$ be such a chain and assume it contains both t_1 and t_2, i.e.

$$l_1 \prec_T \ldots \prec_T l_{j-1} \prec_T l_j = t_1 \prec_T l_{j+1} = t_2 \prec_T l_{j+2} \prec_T \ldots \prec_T l_k .$$

Remark that there cannot be a leaf l_i in between t_1 and t_2 because of Lemma 7.6.5. Lemma 7.6.4 learns us that $l_{j-1} <_{T/t} t <_{T/t} l_{j+2}$, whence $u <_{T/t} u'$. A similar reasoning holds when $(l_i)_i$ contains only one of the children t_1 or t_2. □

We now come to the main theorem for efficiently constructing the order on the leaves during pruning.

Theorem 7.6.8. *Consider a univariate binary tree T and an inner node t of T. Let T/t denote the univariate binary tree we obtain from T by pruning the node t. For all $u, u' \in \widetilde{T/t} \setminus \{t\}$ it holds that*

(i) $u <_{T/t} t \iff (\exists l \in \widetilde{T_t})(u <_T l)$, *and*
$t <_{T/t} u \iff (\exists l \in \widetilde{T_t})(l <_T u)$.

(ii) $u <_{T/t} u' \iff (u <_T u') \vee (u <_{T/t} t <_{T/t} u')$.

Proof.
Pruning the node t can be done in several steps. Take any leaf l of T_t and let $(l = l_1, \ldots, l_k = t)$ be the path in T_t between l and t (i.e. l_{i+1} is the parent of l_i). Now set $T = T_1$ and let T_i ($i = 2, \ldots, k$) be the tree obtained from T_{i-1} by pruning the leaf l_i, i.e. $T_i = T_{i-1}/l_i$. We have that $T_k = T/t$.

(i) We first demonstrate the sufficiency of the statements. If $u <_T l = l_1$, Lemma 7.6.7 tells us that $u <_{T_i} l_i$ for all $i = 2, \ldots, k$, in particular $u <_{T/t} l_k = t$. The case $l <_T u$ is similar. Also remark that there can never at the same time be two leaves $l, l' \in \widetilde{T_t}$ such that $u <_T l$ and $l' <_T u$ since this would lead to $u <_{T/t} t <_{T/t} u$, a cycle.

Now, we proceed with the necessity. If $u <_{T/t} t$, then Lemma 7.6.7 expresses that $u <_{T_{k-1}} t_R$, if t_L, t_R are the children of t with $t_L \prec_{T_{k-1}} t_R$. If $t_R \in \widetilde{T}$, we are done. Otherwise, applying again the same lemma, we find that $u <_{T_{k-2}} t_{RR}$, and we can continue like this until we find $u <_T l$ with $l \in \widetilde{T}$. Similarly for $t <_{T/t} u$.

(ii) We first demonstrate the sufficiency of the statement.

(a) Assume $u <_T u'$. Since both u and u' belong to the sets $\widetilde{T_i} \setminus \{l_i\}$ for $i = 2, \ldots, k$, we can apply Lemma 7.6.7 for the pruning of T_i, $i = 1, \ldots, k-1$, guaranteeing that $u <_{T_i} u'$ for all i, in particular for $i = k$.

(b) If $u <_{T/t} t <_{T/t} u'$, then $u <_{T/t} u'$ by transitivity.

Now we continue with the other direction (necessity). Assume $u <_{T/t} u'$, not $u <_T u'$ and not $u <_{T/t} t <_{T/t} u'$. Because $u \not<_T u'$, we have that $u \not\prec_T u'$, and therefore also $u \not\prec_{T/t} u'$. Hence, there exists some chain (u, t', u') in $<_{T/t}$. By our assumptions, we know that $t' \neq t$, but in that case $t \in \widetilde{T/t} \setminus \{t\}$, so $u <_T u'$, giving rise to conflict with our assumptions. □

Splitting. The following theorem answers the same question for splitting.

Theorem 7.6.9. *Consider a univariate binary tree T and a leaf $t \in \widetilde{T}$. Let $s = (t, c_i, v)$ be a single univariate binary split on T and denote $T(s)$ the tree we obtain from T by splitting the node t into the nodes t_1 and t_2, with $t_1 <_{T(s)} t_2$, using the split s. For all leaves $u, u' \in \widetilde{T(s)} \setminus \{t_1, t_2\}$ (or equivalently, $u, u' \in \widetilde{T} \setminus \{t\}$), the following holds:*

(i) *If $u <_T t$, then $u <_{T(s)} t_2$, $t_1 \not<_{T(s)} u$, $t_2 \not<_{T(s)} u$. Moreover,*

$$\begin{cases} u \prec_{T(s)} t_1 & \text{, if } \begin{cases} (u \prec_T t) \textit{ AND} \\ (a(u)_i \leq_{c_i} v)\,, \end{cases} \\ u <_{T(s)} t_1 & \text{, if } \begin{cases} (u \prec_{T(s)} t_1) \textit{ OR} \\ (\exists u' \in \widetilde{T} \setminus \{t\})((u <_T u') \wedge (u' \prec_{T(s)} t_1))\,, \end{cases} \\ u \parallel_{T(s)} t_1 & \text{, otherwise.} \end{cases}$$

(ii) *If $t <_T u$, then $t_1 <_{T(s)} u$, $u \not<_{T(s)} t_1$, $u \not<_{T(s)} t_2$. Moreover,*

$$\begin{cases} t_2 \prec_{T(s)} u & \text{, if } \begin{cases} (t \prec_T u) \textit{ AND} \\ (b(u)_i >_{c_i} v)\,, \end{cases} \\ t_2 <_{T(s)} u & \text{, if } \begin{cases} (t_2 \prec_{T(s)} u) \textit{ OR} \\ (\exists u' \in \widetilde{T} \setminus \{t\})((t_2 \prec_{T(s)} u') \wedge (u' <_T u))\,, \end{cases} \\ u \parallel_{T(s)} t_2 & \text{, otherwise.} \end{cases}$$

(iii) *If $u \parallel_T t$, then $u \parallel_{T(s)} t_1$ and $u \parallel_{T(s)} t_2$.*
If $u \parallel_T u'$, then $u \parallel_{T(s)} u'$.

(iv) *If $u <_T u'$, then*

$$\begin{cases} u \parallel_{T(s)} u' & \text{, if } \begin{cases} u \not\prec_T u' \textit{ AND} \\ \left[\begin{array}{l}\text{every chain in } \prec_T \text{ connecting } u \\ \text{with } u' \text{ contains } t\end{array}\right. \quad \textit{AND} \\ (u \parallel_{T(s)} t_1) \wedge (u' \parallel_{T(s)} t_2)\,, \end{cases} \\ u <_{T(s)} u' & \text{, otherwise.} \end{cases}$$

The statement

every chain in $\prec_T$ connecting u with u' contains t

is true if and only if

$$(u <_T t <_T u') \wedge (\forall t' \in \widetilde{T})((u <_T t' <_T u') \Rightarrow (t' \not\parallel_T t))\,.$$

Proof.
We start by proving (iv).

(iv) Assume that $u <_T u'$. We distinguish three cases.

(a) $u \prec_T u'$, in which case directly $u <_{T(s)} u'$.

(b) There exist leaves $l_i \in \widetilde{T} \setminus \{t\}$, such that $u \prec_T l_1, \ldots \prec_T l_k \prec_T u'$, implying that also $u \prec_{T(s)} l_1, \ldots \prec_{T(s)} l_k \prec_{T(s)} u'$, and hence $u <_{T(s)} u'$.

(c) The remaining case: $u \not\prec_T u'$ and every chain in $\prec_T$ connecting u with u' contains t. Since $u <_T u'$, there must exist some chain $u <_T t <_T u'$.

- Assume $u <_{T(s)} t_1$. We already know from Lemma 7.6.6 that $t_1 <_{T(s)} u'$, and hence $u <_{T(s)} u$.
- Assume $u \parallel_{T(s)} t_1$. If $t_2 <_{T(s)} u'$, we have $u <_{T(s)} t_2 <_{T(s)} u'$ by Lemma 7.6.6. If $t_2 \parallel_{T(s)} u'$, there is no chain in $\prec_{T(s)}$ connecting u and u'. Indeed, if there were such a chain, then it could not contain t_1 and t_2 (otherwise it wouldn't be a chain). This means the chain would also exist in $\prec_T$ without containing t, a contradiction. So $u \not<_{T(s)} u$.
  Because of Lemma 7.6.7 we already know that $u >_{T(s)} u'$ is not possible (otherwise $u >_T u'$, which on its turn is impossible because $<_{T(s)}$ contains no cycles), therefore the only remaining possibility is that $u \parallel_{T(s)} u'$.

The last equivalence is self-evident. We now proceed with the demonstration of (i). Again, we will show that the cases stated are in fact characterisations.

(i) Assume $u <_T t$. We have $u <_{T(s)} t_2$ because of Lemma 7.6.6. This immediately implies that $t_2 \not<_{T(s)} u$, otherwise we would have a cycle. The same lemma states that either $u <_{T(s)} t_1$ or $u \parallel_{T(s)} t_1$, in both cases leading to $t_1 \not<_{T(s)} u$.

(a) Assume $u \prec_T t$ and $a(u)_i \leq_{c_i} v$. We know that $a(u) \leq_{\mathcal{X}} b(t)$, and $b(t_1)_j = b(t)_j$ for all $j \neq i$, and $b(t_1)_i = v$. Therefore $a(u)_i \leq_{c_i} v$ implies $a(u) \leq_{\mathcal{X}} b(t_1)$, i.e. $u \prec_{T(s)} t_1$.

Conversely, we show that the relation $u \prec_{T(s)} t_1$ only occurs in the case stated. Assume $u \prec_{T(s)} t_1$. Then $a(u) \leq_{\mathcal{X}} b(t_1) \leq_{\mathcal{X}} b(t)$, whence $u \prec_T t$. So, $a(u) \leq_{\mathcal{X}} b(t_1)$, and in particular $a(u)_i \leq_{c_i} b(t_1)_i = v$.

(b) Assume $(\exists u' \in \widetilde{T} \setminus \{t\})((u <_T u') \wedge (u' \prec_{T(s)} t_1))$. Because $u' <_{T(s)} t_1$, we know by Lemma 7.6.4 that $u' <_T t$. This means that we can deduce from (iv) that $u <_T u'$ implies $u <_{T(s)} u'$ (since there exists no chain (u, t, u') in $<_T$). Transitivity now leads to $u <_{T(s)} t_1$.

Conversely, assume $u <_{T(s)} t_1$, but $u \not\prec_{T(s)} t_1$. This means there exists a leaf $u' \in \widetilde{T(s)} \setminus \{t_1, t_2\}$ such that $u <_{T(s)} u' \prec_{T(s)} t_1$. Now

Theorem 7.6.8 learns us that $u <_{T(s)} u'$ implies $u <_T u'$, finishing our demonstration.

(ii) Similar to (i).

(iii) Assume $u \parallel_T u'$. If we would have $u <_{T(s)} u'$, Lemma 7.6.7 would imply that $u <_T u'$, a contradiction. Lemma 7.6.6 provides the proof for the fact that $u \parallel_T t$ implies both $u \parallel_{T(s)} t_1$ and $u \parallel_{T(s)} t_2$. □

Remark 1. This theorem can be easily extended towards multiple splits of the form $s = (L, c, v)$, where L is a chain in $(\widetilde{T}, \leq_T)$. Consider such a chain $(l_i)_{i=1}^k$. The general position of a leaf $u \in \widetilde{T} \setminus L$ w.r.t. to L is (see Figure 7.29):

$$l_1 \leq_T \ldots \leq_T l_{j_1} \leq_T u \leq_T l_{j_2} \leq_T \ldots \leq_T l_k \,,$$

with $j_1 < j_2$, and $u \parallel_T l_i$ for $i = j_1+1, \ldots, j_2-1$. We allow $j_1 = 0$, meaning that $u \leq_T l_{j_2} \leq_T \ldots \leq_T l_k$ and $u \parallel_T l_i$ for $i = 1, \ldots, j_2 - 1$. Similarly, we allow $j_2 = k+1$.

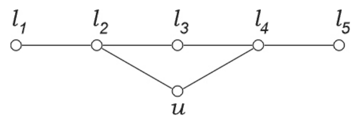


Figure 7.29: The position of a leaf in a chain.

Now we apply the split s. We have that u will also be incomparable to all the children of the l_i, $i = j_1+1, \ldots, j_2-1$. Moreover, because $(l_{j_1})_L \leq_{T(s)} u$, we immediately have $(l_i)_L \leq_{T(s)} u$ for $i = 1, \ldots, j_1$. Likewise, we have $u \leq_{T(s)} (l_i)_R$ for $i = j_2, \ldots, k$. Now, we only need to establish the relationship with the $(l_i)_R$ for $i = 1, \ldots, j_1$, and the $(l_i)_L$ for $i = j_2, \ldots, k$. But since these are two chains, it suffices to find the highest (resp. lowest) i for which $(l_i)_R \leq_{T(s)} u$ (resp. $u \leq_{T(s)} (l_i)_R$), the other relations deducible from them.
It can be checked that we will have $u \parallel_{T(s)} u'$ if and only if $u \parallel_{T(s_i)} u'$, where $s_i = (l_i, c, v)$, for some $i = 1 \ldots, k$. This gives us a way of determining the relation between two leaves $u, u' \in \widetilde{T} \setminus L$.

Remark 2. All lemmas and propositions were formulated on univariate binary trees. However, all the proofs can be simply adapted to hold for univariate trees where the splits are not necessarily binary.

7.7 Future research

Pruning and cross validation. Pruning gets its real power from the derivation of good error and variance estimates from the given training data. Ideally, these estimates can be calculated from a second large data set independent from the training data. However, if no such set is available, a successful technique to resort to is cross validation. The implementation of cross validation in classification trees is, however, not directly adaptable towards ranking trees. But it would surely be a very promising avenue for boosting the accuracy of RT-algorithms.

Another open problem related to pruning is the optimal monotone labelling of the leafs.

Robustness. The problem of non-robustness is one of the most important vices of tree growing procedures. They are highly sensitive to changes in the data set: adding or deleting a few example can cause the choice of a different split. An idea to acquire greater stability would be to create at each split k learning samples $t_{\mathcal{S}}^k$ from $t_{\mathcal{S}}$, and find for each of these k data sets a ranking on the splits. Then these rankings could be aggregated to one final ranking that is then used for choosing the split of t. In that way, a kind of compromise split that performs well on all k samples will be chosen. It can be suspected that incorporating such compromise splits in the growing phase will lead to more stable trees. Of course, the down side is that a lot more computations have to be made, and there is still the problem of how to aggregate the different rankings.

Variants of the algorithm. The main purpose of this chapter was to provide an inkling of how a full-fledged ranking tree algorithm can be developed. The algorithm described in Section 7.5 only applies the most basic and straightforward ideas. Obviously, a lot of more evolved and fine-tuned variants can be thought of:

(i) First of all, instead of simply counting reversed preferences, we could envisage a more advanced splitting measure. Some of our efforts in that direction can be found in [27], where the idea of transforming reversed preference into doubt is incorporated to adapt impurity measures towards ranking.

(ii) Along the same line, we could try to adopt a more regression-like approach, where for example OSDL is applied to obtain the labelling for each split.

(iii) Another degree of freedom that was not further investigated is the choice of the nodes to be split.

(iv) We could also allow more freedom within the multiple split. The algorithm we described simply uses the same split (c, v) for all nodes, but it might be more interesting to let the value of v vary when splitting different leaves.

Optimisation. No real efforts (besides the ones mentioned in Section 7.6) have yet been done to reduce the computational complexity of the algorithm. A lot more research is needed in this direction to cram ranking trees for practical use. One possibility could be to investigate more in depth the impact of closing the relation $\preceq_T$ in a transitive way. This impact is situated on two levels: first of all there is the influence on the relation $\leq_T$ itself (how much do the relations $\preceq_T$ and $\leq_T$ actually differ?), and secondly, we can look at how the tree is affected by it both during splitting and pruning.

Chapter 8

Conclusions and summary

Now the moment has arrived for us to draw our conclusions from everything we've been through together, and hopefully to act upon them in times to come and future days. The journey was sometimes a true ordeal, but in the end worth all the hardness and deprivation.
So, what have we learnt?

8.1 In general

One of the main contributions of this work is the demonstration that investing time in a solid semantical framework is not a waste effort. Although most trail-blazing pioneering work is grounded initially in the firm earth of philosophy and/or semantics, these basics are usually quite rapidly forgotten, and the research is continued in a style where one searches and re-searches among the endless possibilities in combining and extending notions and formulas, only guided by mere logical deduction and a restricted form of trial and error. This dissertation picks up the original threads, spun from philosophical and semantical flocks that were trimmed from different disciplines, and knits them together into a new pattern continuously adding and reassessing the semantical backbone. Like this, we have securely anchored a framework and some adjoined notions for the supervised learning of a ranking. The algorithms that were built starting from these roots prove the efficiency of this non-standard approach.

8.2 Theoretical: syntax and semantics

8.2.1 Classification in supervised learning

Based on simple schemes of functions and relations, it is possible to decompose a classification in the context of the learning problem, resulting in clear representations based on sets or distributions. This then leads to an alternative view on the basic concepts underlying rough sets (the notion of *inconsistency* is better called *doubt*), including extensions based on similarity relations. Also information measures can be quite easily extended toward similarity relations once this framework is adopted.

8.2.2 Ranking in supervised learning

Ranking differs from ordered (ordinal or continuous) classification in the underlying semantics that can be attached to the order, namely that of a (weak) preference relation. The property of monotonicity is due to the use of criteria instead of attributes to describe the objects to be ranked.
When we are confronted with the problem of *reversed preferences* (when an object that is dominated by another object still gets a higher rank) inside the data, the

proposed framework and representations employed for classification deliver a non-invasive solution: the reversed preferences are transformed into *doubt* regarding the preferences, guaranteeing the monotonicity of the resulting representations.

When reversed preference occurs in a distributional context, it is possible to combine the two conflicting probability distributions (via the cumulative distribution functions) and mould them into a single one, freeing the resulting representation of conflicts while respecting the originally given data as much as possible.

These findings form the foundations for the instance-based algorithms OSDL and B-OSDL.

8.2.3 The dominance relation in supervised learning

Extending the dominance relation is no sinecure. The concept is so fundamental that messing around with it tends to inflict unforseen, but severe and possibly unwanted consequences. To get a grip on these consequences, we restricted our generalisation to the context of supervised learning and partitions, resulting in the *partial dominance relation*. As always, we started from semantical considerations and their interactions with the problem.

These findings form the foundations for a model-based algorithm, Ranking Trees.

8.3 Practical: algorithmic

8.3.1 Instance-based

The algorithm OSDL (Ordinal Stochastic Dominance Learner) is an immediate application of the distributional representation theorem for rankings. It stores all data (in a specified format) into memory, and, based on elementary dominance principles, determines a probability distribution for new unseen objects. OSDL contains one parameter, which can be tuned automatically using a leave-one-out cross validation strategy.

A variant on this algorithm, named B-OSDL (Balanced OSDL), probes deeper into the available information. A more refined interpretation of the number of occurrences of reversed preference makes it possible to add more relief to otherwise flat regions of the learned surface. B-OSDL has two parameters to tune.

An extensive experimental setup for learning rankings, based on both artificially generated data and data coming from real surveys shows clearly the supremacy of the OSDL algorithms over other ones. Moreover, they guarantee that the results are monotone (which is not the case for the most competitive algorithm, the Naive Bayes method).

A drawback of instance-based methods is of course that they cannot be interpreted, that they are black boxes.

8.3.2 Model-based

The Ranking Tree algorithm is designed as an answer to this black box criticism. Based on the ideas of classification trees, it provides a tree structure (visualising the rule base), complemented with a lattice-structure on the leaves (visualising the partial dominance structure on the different regions of the data space).

The Ranking Tree algorithm is presented in its most elementary form, making sure that all concepts are applied correctly, and solving all essential algorithmic aspects of the growing of a tree for a ranking problem, like avoiding blind splits and making sure that the final tree is monotone. However, no further research concerning less elementary splitting measures (only a simple counting of reversed preferences is considered) or other heuristics for choosing which leaves to split was conducted. No extensive investigations were done on how to make the algorithm more performing in computational terms, and hence able to deal with larger data sets.

The results of a modest experimental setup with artificial data show that the Ranking Tree algorithm outperforms the other existing tree-based algorithms. But compared to OSDL, no definite conclusion can yet be drawn (except that OSDL is currently able to deal with larger data sets).

The fact that such an elementary and not fine-tuned algorithm does deliver these good results, clearly indicates that this is a very promising avenue for further research.

Samenvatting

Gesuperviseerd rangschikken: van semantiek tot algoritmiek

Dan toch nog een woordje Nederlands in dit boek. In dit korte en bondige na-hoofdstukje zetten we alles nog eens mooi op een rijtje, netjes beginnend met de filosofische beschouwingen uit het eerste hoofdstuk, en eindigend met de conclusies uit het achtste en laatste hoofdstuk.

1 Filosofie en probleemstelling

Rangschikken en ordenen doen we de hele tijd, of het nu gaat om het kopen van een hondje, een sollicitatie, een maatschappelijk onderzoek naar de subjectieve gewaarwording van geluidshinder, de economische evaluatie van faillissementen, of de kwaliteitsopvolging van een proefveld. Continu evalueren we zaken op verschillende criteria om tot een eindbeoordeling te komen. Prefereer je een kleine langharige keffer, of hou je meer van een reusachtige kortharige labrador? Vul je de vacature op met een stille harde werker, of sluit een babbelgraag nauwer aan bij het profiel van de job? Welke soort geluiden leiden ertoe dat “de mensen” het gevoel van geluidsoverlast ervaren, en welke niet?

In deze huidige door de computer overspoelde tijden is het mogelijk om ons voor dergelijke vragen digitaal te laten assisteren, en om de antwoorden te analyseren en zelfs te automatiseren. Aan de hand van een reeks eerder genomen gelijkaardige beslissingen, aan de hand van de uitslagen van een enquête, aan de hand van een reeks observaties tracht men dan de criteria te linken met de eindbeoordeling door middel van een zogenaamd leerproces. Edoch, voor een rangschikking dient het verband tussen de criteria en het eindbeoordeling monotoon, niet-dalend, te zijn.

Een mens is echter niet altijd even consequent en rigoureus, en van een groep mensen kan men al helemaal niet meer verwachten dat ze er allemaal precies dezelfde mening op nahouden, en daar ook naar handelen. Dit leidt soms tot beslissingen die elkaar lijken tegen te spreken (*tegenstrijdige preferenties*). Dit is problematisch voor computers, die moeite hebben zich in te leven in de grillige en subjectieve geesten van ons, mensen.

De grondslag van dit werk is eerder filosofisch, vertrekkende van het besef dat we onze wereld alleen vanuit onze eigen perceptie kunnen waarnemen, en dat we derhalve overeenkomsten nodig hebben om de betekenis die we ergens aan hechten een universeel karakter te geven. In tegenstelling tot de algemeen gangbare overtuiging is dit niet anders met betrekking tot wiskunde en informatica. Een andere interpretatie, een andere context, kortom, een andere semantiek kan leiden tot andere antwoorden tijdens het uitwerken van een wiskundig probleem.

Perceptie en semantiek zijn echter niet de enige leidraden die dit onderzoek in goede banen hebben geleid, en tevens een niet geringe drijfveer vormden. De bedoeling was om steeds het grotere geheel als referentiekader te nemen, om niet te verzanden in de details eigen aan algoritmische benaderingen. Het grotere geheel is echter opgebouwd uit verscheidene onderdelen, welke elk zowel apart als in interactie met elkaar dienen begrepen te worden. Ook een heel belangrijk, zo niet het belangrijkste, opzet was om, eerder dan in het *hoe*, inzicht te verwerven in het *waarom* van de zaken. Tenslotte beoogden we een “non-invasive” houding in onze benadering van het probleem.

2 Een kader voor classificatie

Een classificatie is eigenlijk een afbeelding van een verzameling objecten naar een verzameling van klasselabels. Om daarin wat meer structuur te brengen, worden de objecten beschreven aan de hand van een reeks metingen, zodat elk object met een punt in een vectorruimte overeenkomt. De classificatie zelf is echter gedefinieerd op de verzameling objecten, en niet op de vectorruimte die deze verzameling wiskundig modelleert. Dit is eenvoudig op te lossen door ook de classificatie te modelleren: hecht aan elke vector die overeenkomt met een of meerdere objecten de klasselabels van deze objecten. Dit kan door ofwel de verzameling klasselabels die zo ontstaat als nieuw label te interpreteren, ofwel door een distributie over de klasselabels aan de vectoren te hechten. Het is duidelijk dat als een vector de gelijktijdige representatie is van meerdere objecten die tot verschillende klassen behoren, er niet eenduidig kan gezegd worden tot welke klasse de objecten die door deze vector worden voorgesteld behoren. In dit geval spreken we van *twijfel* tussen de objecten.

Voorgaande kan uitgebreid worden van punten (vectoren) naar hele regionen, waarbij de afspraak geldt dat als (de vectorrepresentatie van) twee objecten tot zo’n gebied in de vectorruimte behoren, er niet voldoende informatie is om deze objecten van elkaar te onderscheiden omdat ze te veel op elkaar lijken. Op deze manier kunnen similariteitsmaten op natuurlijke wijze geïntroduceerd worden in het kader van classificatie.

Op basis van voorgaand kader is het dan niet zo moeilijk meer om informatiematen zoals de Shannon entropie en de Gini diversiteitsindex te veralgemenen voor similariteitsmaten. Ook de onder- en bovenbenaderingen die gebruikt worden in de ruwverzamelingenleer kunnen vanuit dit kader eenvoudig opnieuw opgesteld worden. En als we distributies gebruiken als basis, rollen we recht in het variabele precisie ruwverzamelingenmodel.

3 Overzicht van bestaande rangschikkingsmethodes

Aangezien we de hele tijd rangschikken en ordenen is het niet verwonderlijk dat er in het verleden al een aantal algoritmes werden ontwikkeld voor het probleem van gesuperviseerd rangschikken. Dat dit pas in het laatste decennium gebeurde, en dan zelfs voornamelijk gedurende de laatste jaren, is waarschijnlijk te verklaren door de complexiteit van het probleem. Zowel de basistheorieën en algoritmes omtrent classificatie, als de computers van weleer waren nog niet ver genoeg ontwikkeld om hiermee om te kunnen gaan.

Instantie-gebaseerd. Het enige voorheen gecreeërde instantie-gebaseerde algoritme, OLM (Ordinal Learning Method) [11, 9], dateert reeds uit 1989, en was daarmee het eerste algoritme dat specifiek ontwikkeld werd voor het leren van rangschikkingen. Het boort wel onmiddellijk door naar de essentie en levert tevens de onontbeerlijke monotone resultaten op, iets waarmee in latere algoritmes wel eens wat losser wordt omgesprongen.

Bomen. Een volgende reeks algoritmes concentreert zich specifiek op het groeien van boomstructuren. De auteur van OLM, Ben-David, bijt wederom de spits af in 1995 met MID (Monotone Induction of Decision trees) [10]. Opnieuw bevat de opbouw van zijn algoritme alle basiselementen eigen aan dit specifieke leerprobleem met deze specifieke structuren, maar een stevig gefundeerd kader ontbreekt waardoor dit algoritme aan menige kritiek onderhevig is. Een paar jaar later, besluiten ze in Japan dat Israël lang genoeg solo-slim heeft gespeeld. Makino et al. [74] gooien het over een heel andere boeg, en komen op de proppen met hun boomalgoritmes P-DT, (Positive Decision Tree) and QP-DT (Quasi-monotone P-DT) , specifiek voor het binaire rankschikkingsprobleem. Kort daarop veralgemeende Potharst [89, 90, 91] in Nederland deze aanpak naar niet-binaire problemen, resulterend in de algoritmes MDT (Monotone Decision Tree) en QMDT. Een overzicht van de eigenschappen van al deze boom-algoritmes is terug te vinden in Tabel 3.1 (zie p. 58).

Methodes gebaseerd op ruwverzamelingen (rough sets) Door het gedachtengoed van multicriteria beslissingsanalyse in de noties van ruwverzamelingen te integreren, openden Greco et al. [50, 51, 52, 53] een nieuwe weg in het omgaan met gesuperviseerd rangschikkingen. Hun benadering, DBRS (Dominance-based

Rough Sets approach) kreeg evenwel kritiek te verduren omwille van een dissociatie van de gebruikte maat en de regels die gegenereerd worden. Een andere maat, vrij van deze kritiek, werd daarna voorgesteld door Gediga en Düntsch [48]. Bioch en Popova [15, 16, 87] nemen nog een andere benadering, en baseren zich op de theorie van de Boolese functies die een alternatieve kijk geeft op ruwverzamelingen.

Aggregatiemethodes Ook de Choquet integraal, welke monotoon is van nature, werd al meermaals als uitgangspunt genomen. Verkeyn et al. [116, 117] gebruiken deze rechtstreeks en lieten een genetisch algoritme de vele parameters van deze functie bepalen. TOMASO (Tool for Ordinal Multi-Attribute Sorting and Ordening) van Roubens et al. [76, 96] hanteert een steviger kader voor het gebruik van de Choquet integraal om het probleem van de "commensurabiliteit" (het meetbaar moeten zijn op dezelfde schaal van alle criteria) te omzeilen. De parameters worden berekend via een lineair programma.

Gerelateerde methodes: ordinale classificatie en regressie Hoewel ze niet ontwikkeld werden voor rangschikking, is er nog een hele klasse van algoritmes die wel geordende klassen aankunnen. Hierin dienen alleszins de cumulatieve modellen uit de statistiek vermeld te worden, ontwikkeld door McCullagh [77] in 1980. Ook het veel recentere distributie onafhankelijk model van Herbrich [56, 57, 58] verdient het uitgelicht te worden.

4 Een kader voor rangschikken: elementaire granulatie

Het probleem met de bestaande algoritmes is dat ze allen zonder een algemeen omvattend kader ontwikkeld zijn. Dit maakt dat de onderliggende machinaties soms niet erg duidelijk zijn, of dat ze onbewust bepaalde assumpties aanvaarden die soms wel een grote impact kunnen hebben. Om te beginnen was nog nergens sluitend gedefinieerd wat een rangschikking precies is.

Afgaande op de semantiek van het woord, zoals bepaald in een woordenboek, kunnen we een rangschikking definiëren als een geordende (ordinaal of continu) classificatie waarbij deze orde de betekenis van een (zwakke) preferentierelatie aanneemt. Wanneer de objecten dan beschreven worden door criteria i.p.v. attributen (waarbij criteria geordende attributen zijn met een ordening die overeenkomt met een preferentierelatie) komt op natuurlijke wijze de monotoniteit van het probleem naar boven drijven. Hierbij dient men wel te letten op het feit dat het om de monotoniteit van de gerepresenteerde rangschikking λ_{repr} gaat, en niet om monotoniteit van de gerepresenteerde objecten (de vectoren) en de oorspronkelijke rangschikking λ.

Bij het representeren van een rangschikking komt de semantiek weer op de proppen: als men de verzameling interpretatie zoals bij classificatie wil hanteren moeten de verzamelingen vervangen worden door intervallen, waarop dan weer een nieuwe ordening moet bepaald worden. Afhankelijk van de preferenties van de gebruiker kan deze ordening aangepast worden aan het probleem, maar de meest algemene ordening is wel de klassieke orde $[r_1, r_2] \leq^{[2]} [s_1, s_2] \Leftrightarrow (r_1 \leq s_1) \wedge (r_2 \leq s_2)$. Indien men met probabiliteitsdistributies werkt, treedt de stochastische dominantie op natuurlijke wijze naar voor.

Wanneer we geconfronteerd worden met *tegenstrijdige preferenties* in de data, dan biedt het voorgaande kader en de bijhorende representaties een oplossing die de onaanvaardbare tegenstrijdigheid in de data niet verwijdert maar omzet in aanvaardbare twijfel omtrent de preferenties, welke bovendien de monotoniteit van de oplossing garandeert. In de context van probabiliteitsdistributies kunnen de strijdige distributies gecombineerd worden (door over te gaan op de cumulatieve distributiefuncties), en zelfs samengesmolten worden tot één enkele distributie.

5 OSDL: Ordinale Stochastisch Dominante Leermethode

De voorgaande monotone representaties gebaseerd op distributies (gebaseerd op het cumulatief model) vormen de onmiddellijke basis voor de instantie-gebaseerde algoritmes OSDL en zijn meer geBalanceerde variant B-OSDL. Deze algoritmes memoriseren alle data (in een specifiek formaat), en, gebaseerd op elementaire dominantie principes, wordt dan voor elk object een bijhorende probabiliteitsdistributie gegenereerd. Hiertoe worden in feite evenveel oplossingsoppervlakken geleerd als er klassen zijn. Indien gewenst kan de bekomen distributie ook omgezet worden in één enkel klasselabel. OSDL bevat één parameter, welke automatisch kan afgesteld worden door gebruik te maken van een leave-one-out kruisvalidatie strategie.

De gebalanceerde variant, B-OSDL, graaft dieper in de informatie die latent aanwezig is in de leervoorbeelden. Door een meer verfijnde interpretatie van het aantal tegenstrijdige preferenties, is het mogelijk meer reliëf te creeëren in anders vlakke regionen in de geleerde oppervlakken. B-OSDL bevat twee parameters die afgesteld moeten worden.

Bij een nieuw algoritme hoort uiteraard een uitgebreide experimentele test-fase. We hebben dat dan ook gedaan, en een heleboel andere methodes, zowel voor classificatie (k-nearest neighbour, C4.5, naive Bayes) als voor rangschikking (OLM, MDT, de minimale en maximale extensie), vergeleken met de OSDL-algoritmes. We bekeken zowel artificieel gegenereerde data in verschillende designs, monotoon en niet-monotoon, als data afkomstig uit enquêtes. In zowat alle gevoerde experimenten springen de OSDL-algoritmes eruit, waarbij we nogmaals benadrukken dat OSDL de monotoniteit van de oplossing garandeert (wat niet het geval is voor het meest competitive algoritme, de naïve Bayes).

Een nadeel aan alle instantie-gebaseerde methodes is natuurlijk dat ze de data niet interpreteren, dat het gewoon input-output modellen zijn, zwarte dozen.

6 Een kader voor rangschikken: relationele granulatie

Het kader ontwikkeld in het vierde hoofdstuk is toegespitst op punten (vectoren) in de dataruimte. Er zijn echter een heleboel leermethodes gefundeerd op de partitionering van de dataruimte. Edoch, overgaan van punten naar partities is zeker geen sinecure, want aan de grondslag van de monotoniteit van het probleem ligt de dominantie relatie die enkel op punten is gedefinieerd (monotoniteit komt voort uit het principe van behoud van dominantie). Deze relatie is zo fundamenteel dat elke wijziging vele onvoorziene consequenties tot gevolg kan hebben, sommige heel ingrijpend en mogelijk ongewenst.
Om toch een beter houvast te krijgen op alle mogelijke implicaties, hebben we onze veralgemening beperkt tot de context van partities en gesuperviseerd leren. Vertrekkend van de onderliggende oorspronkelijke semantiek van het basisbegrip, en van de gewenste semantiek voor de gezochte veralgemening (steeds m.b.t. gesuperviseerd leren), leidde onze queeste tot het begrip van *partiële dominantie*. Hiermee gewapend kan dan het principe van behoud van partiële dominantie geformuleerd worden, en een partitie-gebaseerde monotoniteit. De representatiestellingen uit Hoofdstuk 4 kunnen dan vlot omgezet worden van punten naar partities.

7 De basis van rangschikkingsbomen

Boomstructuren voor classificatie zijn sedert hun introductie in de wetenschappelijke wereld altijd zeer populair geweest omwille van hun gemakkelijk interpreteerbare visuele voorstelling van de classificatie. Door hun helderheid en transparantie vormen ze één van de meest intuïtieve antwoorden op het probleem van zwarte doos modellen. Dit laatste hoofdstuk behandelt alle elementaire vragen omtrent de aanpassing van algoritmes voor het groeien van classificatiebomen naar de context van rangschikken, inclusief een aangepaste visuele voorstelling die rekening houdt met de complexere structuur eigen aan rangschikkingen.
Eerst en vooral wordt de meest rudimentaire maat voorgesteld om de keuze van de splitsing van een blad te bepalen: de splitsing die de meeste tegenstrijdige preferenties elimineert wordt gekozen. Maar de meest in het oog springende aanpassing is dat voor rangschikkingsbomen, de blaadjes van de bomen niet meer onafhankelijk zijn, maar verbonden worden via de partiële dominantie relatie. Eerst en vooral leidt dit tot een visuele representatie van de ordening want in dit geval blijkt deze relatie tot een traliestructuur te leiden. Een ander gevolg is dat de volgorde van de blaadjes die gesplitst worden een belangrijke rol gaat spelen in de ontwikkeling van de boom. Een gerelateerd gevolg is dat het splitsen van slechts één enkel blad soms *blind* gebeurt, waarmee we bedoelen dat het algoritme met geen mogelijkheid kan bepalen welke splitsing relevant is. Voor beide problemen dient dus een oplossing en/of heuristiek bedacht te worden. Als een eerste mogelijke optie stellen we voor om twee blaadjes te kiezen, namelijk deze die leiden tot de meeste tegenstrijdige preferenties, en om deze blaadjes gelijktijdig te splitsen (en indien nodig ook alle blaadjes die in de tralie ertussen liggen).

Ook het achteraf snoeien van de boom levert een reeks eigen problemen op. De voorgestelde aanpak is het eenvoudigweg snoeien van de zwakste schakel, waarbij de labels van de blaadjes bepaald worden door een aangepaste vorm van het OSDL algoritme.
Het algoritme dat gepresenteerd wordt is in feite slechts een samenraapsel van alle meest elementaire oplossingen, en is derhalve slechts een basis, een smaakmakertje, voor een volwaardig Ranking Tree algoritme. Ook werd nog niet veel aandacht besteed aan de complexiteit van het algoritme, een paar stellingen omtrent de orde op de blaadjes tijdens het splitsen en snoeien niet te na. Maar desondanks toonden de resultaten van een bescheiden experimentele opzet met artificiële data dat het voorgestelde algoritme andere boomalgoritmes achter zich laat. Een vergelijking met OSDL leidde evenwel nog niet tot finale conclusies (behalve dat OSDL grotere databanken aankan dan het huidige Ranking Tree algoritme).
Het feit dat een dergelijk rudimentair algoritme en totaal niet fijn afgesteld algoritme toch zulke goede resultaten oplevert, is een duidelijk indicatie dat dit voor verder onderzoek een veelbelovende weg is om te bewandelen.

8 Conclusies

De wetenschappelijke resultaten zijn duidelijk zichtbaar: een stevig onderbouwd kader dat zowel classificatie als rangschikking omvat, en daaruit voortvloeiend verscheidene algoritmes, zowel instantie-gebaseerd als model-gebaseerd. Een andere contributie welke zeker niet te onderschatten valt is iets meer aan het zicht onttrokken. Het gaat om de niet echt standaard aanpak van de wiskundige en informatica problemen die aan bod kwamen.
Alhoewel het meeste van het ingrijpende pionierswerk oorspronkelijk geaard is in filosofie en/of semantiek, toch worden deze basisideeën gewoonlijk nogal vlug vergeten. Het onderzoek wordt dan voortgezet als een zoektocht doorheen de oneindige mogelijkheden in het combineren en veralgemenen van begrippen en formules, louter geleid door logische deductie en een beperkte vorm van “trial and error”. Dit proefschrift pikt de originele draden weer op, het garen gesponnen van filosofische en semantische wolvlokken afkomstig uit diverse disciplines, en verweeft de van oudsher vastgelegde semantiek samen met de beoogde semantiek tot een nieuw verfijnd patroon. In plaats van een formeel logisch discours aan de grondslag te leggen, werd het geheel verankerd in semantische overwegingen en filosofische beschouwingen. Dit werk is in zijn geheel een demonstratie van de kracht van een dergelijke aanpak, het toont aan dat het investeren in een gedegen semantisch kader absoluut geen tijdverspilling is.

Coda

SUNDAY, NOVEMBER 9, 2003. *Well people, that was it. Time to unfold and close the curtain on this rather intense stage of my life. Time to rest and enjoy the thrill, to relish in the knowledge that this four year during performance ended with an extremely happy note, both professionally in the recognition I received from this work, and personally in the fabulous dreamgirl I met at exactly the right moment. Time to let me sweep away by the current of the next and very likely even more intense and gratifying chapter of the story that makes up my life.*
Only one final rush still separates me from the ultimate last and grand finale of this Ph.D., only two weeks left to finish (actually only 12 days) my swirling and maybe a bit unusual presentation. I hope my estimations have become a bit more reliable of late, because I have to admit – a bit reluctantly – that I still have to start from scratch and with the crazy ideas I wish to realise, time is definitely not on my side (hey, that's something new! Where did I hear that one before?)

WEDNESDAY, NOVEMBER 12, 2003. *Hard to believe, but I've done it again. I still have practically nothing for my presentation. But, I just – its 5am– finished my cover design. I'm quite happy with it, not bad for a couple of hours intensive struggle, certainly for somebody who never worked in Photoshop before. Today I also learned how to use an animation program for all my special effects, and later on, I will try to understand the basics of some video editing program. I think I'm crazy. Any sane person would say the task is sheer impossible, but you know, positive stress can be very stimulating, and the sheer impossible is just the kind of tantalising challenge I adore.*

THE REMAINING DAYS. *Well, I won't be able to keep you up to date with my final progressions, because this manuscript has to go to the printing office. Today, now. So, that's it. I really enjoyed talking to you, and hope you enjoyed every bit as much as I. Take care.*

THE FINAL WORDS. *Bernard, because some things can not be repeated often enough: thank you for everything.*

Bibliography

[1] J. Aczél and Z. Daróczy, *On Measures of Information and Their Characterizations*. New York, NY: Academic Press, 1975.

[2] A. Agresti, *An Introduction to Categorical Data Analysis*. New York: John Wiley & Sons, 1996.

[3] B. Apolloni, G. Zamponi, and A. Zanaboni, "Learning fuzzy decision trees," *Neural Networks*, vol. 11, no. 5, pp. 885–895, 1998. [Online]. Available: http://citeseer.nj.nec.com/apolloni96learning.html

[4] K. Arrow, *Social Choice and Individual Values*, 1st ed. New York, NY: John Wiley, 1951.

[5] C. Bana e Costa and J. Vansnick, *The MacBeth approach: basic ideas, software and an application*. Dordrecht: Kluwer Academic Publishers, 1999, pp. 131–157.

[6] R. Barletta, "A hybrid indexing and retrieval strategy for advisor cbr systems built with remind," in *Proceedings of the 2nd European Workshop on CBR (EWCBR94)*. AcknoSoft press, 1994, p. 4958.

[7] J. Bazan, "A comparison of dynamic and non-dynamic rough set methods for extracting laws from decision tables," in *Rough Sets in Knowledge Discovery*, L. Polkowski and A. Skowron, Eds. Heidelberg: Physica-Verlag, 1998, ch. 19, pp. 321–365. [Online]. Available: http://citeseer.nj.nec.com/328976.html

[8] G. Beliakov, "Monotone approximation of aggregation operators using least squares splines," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 6, pp. 659–676, dec 2002.

[9] A. Ben-David, “Automatic generation of symbolic multiattribute ordinal knowledgebased dss: Methodology and applications,” *Decision Sciences*, vol. 23, pp. 1357–1372, 1992.

[10] ——, “Monotonicity maintenance in information-theoretic machine learning algorithms,” *Machine Learning*, vol. 9, pp. 29–43, 1995.

[11] A. Ben-David, L. Sterling, and Y. Pao, “Learning and classification of monotonic ordinal concepts,” *Computational Intelligence*, vol. 5, pp. 45–49, 1989.

[12] R. Benbenishty and W. Chen, “Decision making by the child protection team in sheeba hospital,” 2003, in press.

[13] R. Bender and A. Benner, “Calculating ordinal regression models in SAS ans S-Plus,” *Biometrical Journal*, vol. 42, no. 6, pp. 677–699, 2000. [Online]. Available: http://wwwhomes.uni˙bielefeld.de/rbender/biblio.htm

[14] P. Berkhin, “Survey of clustering data mining techniques,” Acrue Software, Inc., San Jose, CA, research paper, 2002. [Online]. Available: http://www.accrue.com/products/researchpapers.html

[15] J. Bioch and V. Popova, “Rough sets and ordinal classifcation,” in *Algorithmic Learning Theory*, ser. Notes in Artificial Intelligence, A. Arimura and S. Jain, Eds. Springer, 2000, vol. 1968, pp. 291–305.

[16] ——, “Bankruptcy prediction with rough sets,” Erasmus University Rotterdam, Technical report ERS-2001-11-LIS, 2001. [Online]. Available: http://ideas.repec.org/s/dgr/eureri.html

[17] ——, “Labeling and splitting criteria for monotone decision trees,” in *Proceedings of the 12th Belgian-Dutch Conference on Machine Learning (BENELEARN’2002)*, M. Wiering, Ed., Utrecht, 2002, pp. 3–10.

[18] ——, “Monotone decision trees and noisy data,” in *Proceedings of the 14th Belgium-Dutch Conference on Artificial Intelligence (BNAIC’2002)*, H. Blockeel and M. Denecker, Eds., Leuven, 2002, pp. 19–26.

[19] G. Birkhoff, *Lattice Theory*. Providence, Rhode Island: Amer. Math. Soc. Coll. Publ., 1973, vol. 25.

[20] D. Bouyssou, “Building criteria: a prerequisite for MCDA,” in *Readings in Multiple Criteria Decision Aid*, C. Bana e Costa, Ed. Heidelberg: Springer-Verlag, 1990, pp. 58–80.

[21] X. Boyen and L. Wehenkel, “Automatic induction of fuzzy decision trees and its application to power system security assessment,” *Fuzzy Sets and Systems*, vol. 102, pp. 3–19, 1999.

[22] L. Breiman, “Using convex pseudo-data to increase prediction accuracy,” Statistics Department University of California, Berkely, Technical Report 513, 1998.

[23] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. New York, NY: Chapman & Hall, 1984.

[24] L. Breslow and D. Aha, “Simplifying decision trees: a survey,” *Knowledge Engineering Review*, vol. 1, pp. 1–40, 1997. [Online]. Available: http://www.aic.nrl.navy.mil/˜aha/pub-details.html#articles

[25] C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998. [Online]. Available: http://mplab.ucsd.edu/pub/readings/Burges98.pdf

[26] K. Cao-Van and B. De Baets, “A decomposition of k-additive Choquet and k-maxitive Sugeno integrals,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 2, pp. 127–143, 2001.

[27] ——, “Impurity measures for ranking problems,” in *Proceedings of Eurofuse 2002*, B. De Baets, J. Fodor, and G. Pasi, Eds., 2002, pp. 285–290.

[28] G. Cattaneo, “Abstract approximation spaces for rough theories,” in *Rough Sets in Knowledge Discovery 1, Methodology and Applications*, L. Polkowski and A. Skowron, Eds. Heidelberg: Physica-Verlag, 1998, pp. 59–98.

[29] M. Chmielewski, J. Grzymala-Busse, N. Peterson, and S. Than, “The rule induction system LERS – a version for personal computers,” *Foundations of Computing and Decision Sciences*, vol. 18, pp. 181–212, 1993.

[30] G. Choquet, “Theory of capacities,” *Ann. Institute Fourier*, vol. 5, pp. 131–295, 1953.

[31] N. Cliff, “Prediciting ordinal relations,” *British Journal of Mathematical and Statistical Psychology*, vol. 47, pp. 127–150, 1994.

[32] C. Cortes and V. Vapnik, “Support vector networks,” *Machine Learning*, vol. 20, pp. 273–297, 1995.

[33] B. Davey and H. Priestly, *Introduction to Lattices and Order*, 2nd ed. Cambridge, U.K.: Cambridge University Press, 2002.

[34] W. De Keyser, “Handleiding argus-software,” 2003.

[35] W. De Keyser and P. Peeters, *ARGUS - a new multiple criteria method based on the general idea of outranking*, ser. EUROCOURSES: Environmental Management. Dordrecht: Kluwer Academic Publishers, 1994, vol. 3, pp. 263–278.

[36] J. Deogun, V. Raghavan, and H. Sever, “Exploiting upper approximation in the rough set methodology,” in *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, U. Fayyad and R. Uthurusamy, Eds. AAAI Press, 1995, pp. 69–74. [Online]. Available: http://citeseer.nj.nec.com/deogun95exploiting.html

[37] Y. DeSmet, “Butterfly auctions: clustering the bidding space,” 2003, accepted for the Proceedings of the 6th International Conference on Electronic Research (ICECR6 2003).

[38] D. Dubois, W. Ostasiewicz, and H. Prade, “Fuzzy sets: history and basic notions,” in *Fundamentals of Fuzzy Sets*, D. Dubois and H. Prade, Eds. Boston: Kluwer Academic Publishers, 2000, pp. 21–124.

[39] P. Fishburn, *Interval Orders and Interval Graphs*. John Wiley, 1985.

[40] ——, “Generalizations of semiorders: A review note,” *Journal of Mathematical Psychology*, vol. 41, pp. 357–366, 1997.

[41] J. Fodor and M. Roubens, *Fuzzy Preference Modelling and Multicriteria Decision Support*. Kluwer Academic Publishers, 1994.

[42] E. Frank and M. Hall, “A simple approach to ordinal classification,” *Lecture Notes in Computer Science*, vol. 2167, pp. 145–56, 2001. [Online]. Available: http://www.cs.waikato.ac.nz/˜ml/publications/2001/ordinal˙tech˙report.pdf

[43] Y. Ganzach, “Goals as determinants of nonlinear noncompensatory judgment strategies,” *Organizational Behavior and Human Decision Processes*, vol. 56, pp. 422–440, 1993.

[44] Y. Ganzach, Y. Kluger, and N. Kleiman, “Making decisions from an interview: Expert measurement and mechanical combination,” *Personnel Psychology*, vol. 53, pp. 1–20, 2000. [Online]. Available: http://recanati.tau.ac.il/faculty/pdf/yoav/exmesr6.doc

[45] G. Gediga and I. Düntsch, *Rough set data analysis, A road to non-invasive knowledge discovery*. Bangor, UK: Methoδos, 2000.

[46] ——, “Statistical techniques for rough set data analysis,” in *Rough Set Methods and Applications*, L. Polkowski, S. Tsumoto, and T. Y. Lin, Eds. Heidelberg: Physica Verlag, 2000, pp. 545–565.

[47] ——, “Roughian - rough information analysis,” *International Journal of Intelligent Systems*, vol. 46, pp. 121–147, 2001. [Online]. Available: http://www.cosc.brocku.ca/˜duentsch/papers/roughian.html

[48] ——, “Approximation quality for sorting rules,” *Computational Statistics and Data Analysis*, vol. 40, no. 3, pp. 499–526, 2002.

[49] M. Grabisch, “k-order additive discrete fuzzy measures and their representation,” *Fuzzy Sets and Systems*, vol. 92, pp. 167–189, 1997.

[50] S. Greco, B. Mattarazo, and S. Słowiński, “A new rough set approach to evaluation of bankruptcy risk,” in *Operational Tool of the Management of Fincancial Risks*, C. Zopounidis, Ed. Dordrecht: Kluwer, 1998, pp. 121–136.

[51] ——, “Multicriteria classification by dominance-based rough set approach, methodological basis of the 4emka system.” 2000. [Online]. Available: http://www-idss.cs.put.poznan.pl/4emka/

[52] ——, “An algorithm for induction of decision rules consistent with the dominance principle,” in *Rough Sets and Current Trends in Computing*, ser. Lecture Notes in Computer Science, Z. Wojciech and Y. Yao, Eds., vol. 2005. Springer, 2001.

[53] ——, “Rough set theory for multicriteria decision analysis,” *European Journal of Operational Research*, vol. 129, no. 1, pp. 1–47, February 2001.

[54] S. Gunn, “Support vector machines for classification and regression,” Faculty of Engineering and Applied Science, Department of Electronics and Computer Science,” Technical report, 1998.

[55] R. Hartley, “Transmission of information,” *Bell Systems Technical Journal*, vol. 7, pp. 535–563, 1928.

[56] R. Herbrich, T. Graepel, and K. Obermayer, “Regression models for ordinal data: a machine learning approach,” Technical University of Berlin, Tech. Rep. TR 99-3, 1999. [Online]. Available: http://research.microsoft.com/users/rherb/techreports.htm

[57] ——, “Large margin rank boundaries for ordinal regression,” in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA: MIT Press, 2000, pp. 115–132. [Online]. Available: http://research.microsoft.com/users/rherb/journals.htm

[58] R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Obermayer, “Learning preference relations for information retrieval,” in *Proceedings of the AAAI Workshop Text Categorization and Machine Learning, International Conference on Machine Learning*, Madison, 1998, pp. 80–84. [Online]. Available: citeseer.nj.nec.com/herbrich98learning.html

[59] E. Hernández and J. Recasens, “A reformulation of entropy in the presence of indistinguishability operators,” *Fuzzy Sets and Systems*, vol. 128, pp. 185–196, aug 2002.

[60] B. Hjørland, “The classification of psychology: A case study in the classification of a knowledge field,” *Knowledge Organization*, vol. 25, no. 4, pp. 162–201, 1998.

[61] T. Joachims, “Optimizing search engines using clickthrough data,” in *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2002. [Online]. Available: citeseer.nj.nec.com/joachims02optimizing.html

[62] M. Joshi, G. Karypis, and V. KumarJoshi, “ScalparC: A new scalable and efficient parallel classification algorithm for mining large datasets,” in *IPPS: 11th International Parallel Processing Symposium*. IEEE Computer Society Press, 1998. [Online]. Available: citeseer.nj.nec.com/joshi98scalparc.html

[63] A. Knobbe, H. Blockeel, A. Siebes, and D. van der Wallen, “Multi-relational data mining,” Centrum Wisk. Inform., Amsterdam, Tech. Rep. INS-R9908, may 1999. [Online]. Available: http://www.cs.kuleuven.ac.be/cgi-bin-dtai/publ˙info.pl?id=20404

[64] A. J. Knobbe, A. Siebes, and D. van der Wallen, “Multi-relational decision tree induction,” in *Proceedings of PKDD ’99, Prague, Czech Republic*, 1999.

[65] J. Komorowski, Z. Pawlak, L. Polkowski, and A. Skowron, “Rough sets: a tutorial,” in *Rough Fuzzy Hybridization. A new trend in decision-making*, S. Pal and A. Skowron, Eds. Singapore: Springer, 1999, pp. 3–98. [Online]. Available: http://citeseer.nj.nec.com/komorowski98rough.html

[66] S. Kramer, G. Widmer, B. Pfahringer, and M. De Groeve, “Prediction of ordinal classes using regression trees,” *Fundamenta Informaticae*, vol. 47, no. 1-2, pp. 1–13, 2001. [Online]. Available: http://www.cs.waikato.ac.nz/˜ml/publications/2001/ordinal˙regression2001.pdf

[67] S. Kramer, “Relational learning vs. propositionalization, investigations in inductive logic programming and propositional machine learning,” Ph.D. dissertation, Technischen Universität Wien, sep 1999.

[68] E. Lehmann, *Nonparametrics, Statistical Methods based on Ranks*. New Jersey: Prentice Hall, 1998.

[69] H. Leiva and V. Honavar, “Experiments with mrdtl – a multi-relational decision tree learning algorithm,” in *Proceedings of the Workshop on Multi-Relational Data Mining (MRDM-2002)*, S. Džeroski, L. D. Raedt, and S. Wrobel, Eds. University of Alberta, Edmonton, Canada, jul 2002, pp. 97–112. [Online]. Available: http://www-ai.ijs.si/SasoDzeroski/MRDM2002/#papers

[70] L. Lerner, “Fake history that is flatly wrong,” *The Textbook Letter*, vol. 2, no. 6, 1992. [Online]. Available: http://www.textbookleague.org/26flat.htm

[71] W. Loh and Y. Shih, “Split selection methods for classification trees,” *Statistica Sinica*, vol. 7, pp. 815–840, 1997.

[72] R. Luce, “Several unresolved conceptual problems of mathematical psychology,” *Journal of Mathematical Psychology*, vol. 41, pp. 79–87, 1997.

[73] B. Magee, *Bekentenissen van een filosoof (original title: Confessions of a Philosopher)*. Amsterdam: Flamingo Pockets, 1997.

[74] K. Makino, T. Suda, H. Ono, and T. Ibaraki, “Data analysis by positive decision trees,” *IEICE Trans. Inf. and Syst.*, pp. 76–88, 1999. [Online]. Available: http://www.ee.psu.ac.th/ieice/1999/files/e000d01.htm

[75] J. Marichal, “Aggregation operators for multicriteria decision aid,” Ph.D. dissertation, University of Liège, Liège, Belgium, 1998. [Online]. Available: http://www.math.byu.edu/˜marichal/

[76] J. Marichal, P. Meyer, and M. Roubens, “On a sorting procedure in the precence of qualitative interacting points of view,” sep 2002, submitted to Elsevier Science. [Online]. Available: http://www.math.byu.edu/˜marichal/Mywebpage/internetfiles/TOMASO.pdf

[77] P. McCullagh, “Regression models for ordinal data,” *Journal of the Royal Statistical Society – Series B*, vol. 42, pp. 109–142, 1980. [Online]. Available: http://galton.uchicago.edu/˜pmcc/publications.html

[78] P. McCullagh and J. Nelder, *Generalized Linear Models*. New York: Chapmann and Hall, 1989.

[79] S. Murthy, “On growing better decision trees from data,” Ph.D. dissertation, The Johns Hopkins University, Baltimore, Maryland, 1997. [Online]. Available: http://citeseer.nj.nec.com/article/murthy97growing.html

[80] ——, “Automatic construction of decision trees from data: A multi-disciplinary survey,” *Data Mining and Knowledge Discovery*, vol. 2, no. 4, pp. 345–389, 1998. [Online]. Available: http://citeseer.nj.nec.com/article/murthy97growing.html

[81] H. Nagesh, S. Goil, and A. Choudhary, “Adaptive grids for clustering massive data sets,” in *(online) Proceedings of the First SIAM International Conference on Data Mining (SDM01)*, 2002. [Online]. Available: http://www.siam.org/meetings/sdm01/pdf/sdm01˙07.pdf

[82] T. Nguyẽn, “Scalable algorithms for learning large decision trees,” Ph.D. dissertation, School of Information Science, Japan Advanced Institute of Science and Technology, Japan, jan 2001.

[83] W. Ogryczak and Ruszczyński, “Dual stochastic dominance and related mean-risk models,” *SIAM Journal of Optimisation*, vol. 13, pp. 60–78, 2002.

[84] Z. Pawlak, “Information systems: theoretical foundations,” *Information Systems*, vol. 6, no. 3, pp. 205–218, 1981.

[85] ——, “Rough sets,” *International Journal of Computer and Information Science*, vol. 116, pp. 341–356, 1982.

[86] M. Pirlot and P. Vincke, *Semiorders: properties, respresentations, applications*, ser. Theory and Decision Library, Series B: Mathematical and Statistical Methods. Kluwer Academic Publishers, 1997.

[87] V. Popova, “Knowledge discovery and monotonicity (draft title),” September 2003, first draft of Ph.D. dissertation, Erasmus University Rotterdam.

[88] R. Potharst, “Classification using decision trees and neural networks,” Ph.D. dissertation, Erasmus University Rotterdam, 1999.

[89] R. Potharst and J. Bioch, “Decision trees for ordinal classification,” *Intelligent Data Analysis*, pp. 97–112, 2000.

[90] R. Potharst, J. Bioch, and R. van Dordregt, “Quasi-monotone decision trees for ordinal classification,” Erasmus University Rotterdam, Tech. Rep. EUR-FEW-CS-98-01, 1998.

[91] R. Potharst and A. Feelders, “Classification trees for problems with monotonicity constraints,” *SIGKDD Explorations*, vol. 4, no. 1, pp. 1–10, 2002.

[92] J. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, pp. 81–106, 1986.

[93] ——, *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.

[94] A. Rényi, *Probability Theory*. Amsterdam: North-Holland, 1970.

[95] S. Romanski, “Operations on families of sets for exhaustive search given a monotonic function,” in *Proceedings of the 3rd International Conference on Data and Knowledge Bases*, C. Beeri, J. Smith, and U. Dayal, Eds., Jerusalem, Israel, 1998, pp. 28–30.

[96] M. Roubens, “Ordinal multiattribute sorting and ordering in the presence of interacting points of view,” in *Aiding Decisions with Multiple Criteria: Essays in Honor of Bernard Roy*, D. Bouyssou, E. Jacquet-Lagrèze, P. Perny, R. Słowiński, D. Vanderpooten, and P. Vincke, Eds. Dordrecht: Kluwer Academic Publishers, 2001, pp. 229–246.

[97] B. Roy, *Multicriteria Methodology for Decision Aiding*. Dordrecht: Kluwer Academic Publishers, 1996.

[98] C. Schaffer, “Overfitting avoidance as bias,” *Machine Learning*, vol. 10, pp. 153–178, 1993. [Online]. Available: http://wwwcs.hunter.cuny.edu/faculty/schaffer/papers/list.html

[99] L. Schumaker, “Triangulation methods,” in *Topics in Multivariate Approximation*, C. Chui, L. Schumaker, and F. Utreras, Eds. New York: Academic Press, 1987, pp. 219–232.

[100] G. Shah Hamzei and D. Mulvabey, “On-line learning of fuzzy decision trees for global path planning,” *Engineering Applications of Artificial Intelligence*, vol. 12, pp. 93–109, 1999.

[101] C. Shannon, “A mathematical theory of communiction,” *Bell Systems Technical Journal*, vol. 27, pp. 379–423, 1948.

[102] M. Shapira and R. Benbenishty, “Modeling judgements and decisions in cases of alleged child abuse and neglect,” School of Social Work, Hebrew University of Jerusalem, Israel, Working paper, 1991.

[103] M. Shapira, “Decision making by probation offciers: A theoretical model and its practical implementation as a decision aid,” Paul Baerwald School of Social Work, Jerusalem, Israel, Monograph, 2000.

[104] A. Shashua and A. Levin, “Taxonomy of large margin principle algorithms for ordinal regression problems,” Leibniz Center for Research, School of Computer Science and Eng., the Hebrew University of Jerusalem, Technical report 2002-39, 2002. [Online]. Available: http://www.cs.huji.ac.il/˜shashua/papers/k-planes-long.pdf

[105] Y. Shih, “Selecting the best splits for classifcation trees with categorical variables,” *Statistics and Probability Letters*, vol. 54, pp. 341–345, 2001. [Online]. Available: http://www.math.ccu.edu.tw/˜yshih/paper.html

[106] D. Ślezak, “Approximate reducts in decision tables,” in *Proceedings of the Sixth International Conference, Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU’96)*, vol. 3, 1996, pp. 1159–1164. [Online]. Available: http://citeseer.nj.nec.com/slezak96approximate.html

[107] R. Słowiński and D. Vanderpooten, “Similarity relation as a basis for rough approximations,” in *Advances in Machine Intelligence and Soft Computing 4*, P. Wang, Ed., 1997, pp. 17–33.

[108] ——, “A generalized definition of rough approximations based on similarity,” *IEEE Transactions on Data and Knowledge Engineering*, vol. 12, no. 2, pp. 331–336, 2000.

[109] J. Stefanowski, “On rough set based approaches to induction of decicion rules,” in *Rough Sets in Data Mining and Knowledge Discovery*, S. S. Polkowski L., Ed. Heidelberg: Physica-Verlag, 1998, pp. 500–529.

[110] S. Stevens, “Measurement, psychophysics and utility,” in *Measurement: Definitions and Theories*, C. Churchman and P. Ratoosh, Eds. New York: John Wiley, 1959, ch. 2.

[111] A. Suárez and J. Lutsko, "Globally optimal fuzzy decision trees for classification and regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, pp. 1297–1311, 1999.

[112] M. Sugeno, "Theory of fuzzy integrals and its applications," Ph.D. dissertation, Tokyo Institute of Technology, 1974.

[113] L. Torgo, "Inductive learning of tree-based regression models," Ph.D. dissertation, Departament of Computer Science, Faculty of Science, Universidade do Porto, Portugal, sep 1999. [Online]. Available: http://www.liacc.up.pt/˜ltorgo/PhD/

[114] A. Tversky, "Features of similarity," *Psychological Review*, vol. 84, pp. 327–352, 1977.

[115] V. Vapnik, *Statistical Learning Theory*. New York: John Wiley and Sons, 1998.

[116] A. Verkeyn, D. Botteldooren, B. De Baets, and G. De Tré, "Modelling annoyance aggregation with Choquet integrals," in *Proceedings of the Eurofuse Workshop on Information Systems*, B. De Baets, J. Fodor, and G. Pasi, Eds., 2002, pp. 259–264.

[117] ——, "Sugeno integrals for the modelling of noise annoyance aggregation," *Lecture Notes in Computer Science*, vol. 2715, pp. 277–284, 2003.

[118] P. Vincke, "Basic concepts of preference modelling," in *Readings in Multiple Criteria Decision Aid*, C. Bana e Costa, Ed. Heidelberg: Springer-Verlag, 1990, pp. 101–118.

[119] X. Wang, B. Chen, G. Qian, and F. Ye, "On the optimization of fuzzy decision trees," *Fuzzy Sets and Systems*, vol. 112, pp. 117–125, 2000.

[120] D. Watson, *nngridr: An Implementation of Natural Neighbor Interpolation*. Claremont, Australia: Dave Watson Publisher, 1994.

[121] P. Watzlawick, J. Beavin, and D. Jackson, *De pragmatische aspecten van de menselijke communicatie (original title: Pragmatics of Human Communications)*. Deventer: van Loghum Slaterus, 1970.

[122] *Webster's Encyclopedia Unabridged Dictionary of the English Language*. New York: Gramercy Books, 1989.

[123] N. Wiener, *Cybernetics, or Control and Communication in the Animal and the Machine*. Cambridge: Technology Press MIT, 1948.

[124] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, oct 1999. [Online]. Available: www.cs.waikato.ac.nz/˜ml/weka/book.html

[125] R. Yager, “On ordered weighted averaging operators in multicriteria decision making,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 18, pp. 183–190, 1988.

[126] W. Ziarko, “Variable precision rough sets model,” *Journal of Computer and System Sciences*, vol. 46, pp. 39–59, 1993.

The author, Kim Cao-Van, was born in 1976
as the son of a mathematician and a psychologist.
He graduated in 1998 in pure mathematics.
Afterwards, his interest was captured by the more applied fields of multicriteria decision aid and machine learning.
This unusual and personal manuscript is the result of four years of research and struggle of an innovative mind.

"Even the most inattentive reader will be touched immediately, and the logically and mathematically oriented reader will in addition be immediately charmed, by the rigorous, utmost precise formulation and analysis of both the semantical aspects of the ranking problem and the algorithmic aspects which give a theoretical foundation to several solution methods, and their implementation with the treatment of actual problems in mind."

UNIVERSITEIT GENT