Preference Inference Based on Maximising Margin

Mojtaba Montazery Hedeshi
MS IN COMPUTER ENGINEERING (AI AND ROBOTICS)
114220093

Thesis submitted for the degree of Doctor of Philosophy

NATIONAL UNIVERSITY OF IRELAND, CORK

FACULTY OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

INSIGHT CENTRE FOR DATA ANALYTICS

July 2018

Head of Department: Prof. Cormac Sreenan

Supervisors: Dr. Nic Wilson
Prof. Barry O'Sullivan

Research supported by Science Foundation Ireland (SFI)

Contents

List of Figures

List of Tables

I, Mojtaba Montazery Hedeshi, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

Mojtaba Montazery Hedeshi

To my wife, Fatemeh Amrollahi, my parents, Akbar Montazery and Fatemeh Zeinali, my siblings, Mohammad Javad Montazery and Maryam Montazery, and my little lovely daughter Kowthar.

Acknowledgements

First of all, I would like to thank God, the Almighty for the continuous endless overflowing blessings upon me. Also, this research would have not been possible without the tremendous amount of unconditional support that was received from various sources. I would like to seize this opportunity to express my sincere gratitude and would like to thank some of them in particular.

Firstly, I would like to thank my primary supervisor Nic Wilson for all the guidance, support and patience, from the very beginning of this journey right to the very end – I appreciate all and everything that you have done for me during the last few years. I would also like to acknowledge and thank my co-supervisor Barry O'Sullivan, and also Science Foundation Ireland for the financial support (Grant No. SFI/12/RC/2289).

I would like to thank all the reviewers of my papers submitted to different conferences and journals, for the constructive criticism and pointers that helped me improve and write better papers. I would like to thank my external examiner Patrice Perny and my internal examiner Steven Prestwich for their comments and feedback and for the enjoyable discussion during my viva voce.

I would like to thank the Insight lab admin office, Eleanor, Caitríona and Linda for all their help handling all of the administration of the thesis, and also the Insight IT department Peter for handling all my IT needs. I am also grateful to all current and previous members of the Insight, with whom I had many productive and enjoyable discussions.

I owe my parents, my wife and friends a debt of gratitude for their permanent moral support and prayers. Undoubtedly this work would have not been completed without their unconditional help.

Abstract

In a decision-making problem, where we need to choose a particular decision from a set of possible choices, the user often has some preferences which determine if one decision is preferred over another. When the number of choices is large, an intelligent system can help the user by attempting to learn user preferences. One way of learning user preferences is based on the maximum margin approach, where maximising the margin can be seen as satisfying each existing preference input to the greatest degree. In this thesis, we first apply this method to a real-world application, ride-sharing, and examine its potential effectiveness.

Nevertheless, we show that the maximum margin preference learning approach is sensitive to the way that preferences inputs and features are scaled. We explain why it is naturally expected that a preference relation is scaling invariant, and go on to construct and characterise some preference relations that are invariant to the scaling of (i) preferences inputs, (ii) features, and (iii) both preferences inputs and features simultaneously. We compare these relations and propose two algorithms to find the optimal elements according to each relation.

In the last main chapter, we argue that the rescaling of features is also an issue in the standard SVM classification and propose a new form of more conservative classification that is invariant to the rescaling of features. We argue that this cautious way of classification could be helpful in some critical decision-making applications.

Chapter 1

Introduction

1.1 Introduction

Choosing a decision among a set of alternatives is a recurrent task in our daily lives. In these situations, user preferences play a major role in directing individuals to make decisions. Preference reasoning is a multi-disciplinary topic that has been extensively studied in economics, psychology, philosophy, logics and other human-centred disciplines. In the context of Artificial Intelligence (e.g., constraint satisfaction, planning, search, resource allocation and electronic commerce), an intelligent reasoning mechanism learns this available preferential information in order to form a *preference relation* on the set of alternatives. Then, we might be able to *prefer* one to another regarding the induced preference relation. The relation will be *total* if we can compare every pair of alternatives.

Though the notion of preferences sounds simple, working with them can be a very difficult task. There are a number of reasons for this, the most obvious one being the cognitive difficulty of specifying preferences. The degree of desirability that is inherent to preferences is most commonly represented either quantitatively by means of utility functions for instance, or qualitatively by means of pairwise comparisons. Preferences can be stated *explicitly* or *implicitly*. Explicit preferences are directly stated by a decision maker. Then, any generated outcome has to comply with all the explicit preferences. Implicit preferences are indirectly discovered from the available information about the user.

The maximum margin based preference relation is an approach that can be effectively used in many applications, as we have done in ride-sharing. This method leads to a total relation by maximising the margin, where maximising the margin can be interpreted as satisfying each existing preference statement to the greatest degree.

For any preference relation, we would often expect that the relation is invariant to the rescaling of preference inputs; e.g., if we know that the user prefers a car with feature vector a to a car with feature vector b, we naturally expect that $2a$ is also preferred to $2b$. However, this is not the case for the maximum margin approach, and rescaling of preferences inputs can change the resultant preference relation. In addition to the rescaling of inputs, different ways of normalization of features (i.e., scaling the domain of each feature)—which is a fundamental pre-processing stage for any maximum margin based method—can lead to completely different relations.

The main focus of this thesis is to look at some more cautious preference relations that are invariant to the rescaling of (i) preferences inputs, (ii) features, and (iii) both preferences inputs and features simultaneously. We investigate the effect of rescaling and characterise some rescaling-invariant preference relations.

1.2 Overview and Contribution

In this section, we give an outline of the main chapters of the thesis, along with highlighting the main contributions.

Chapter 2: Background

This chapter introduces some definitions and the related formalism so as to aid the understanding of the main chapters. The discussions of this chapter are twofold. First, we include some related definitions and results about convex sets and convex cones. This will be particularity helpful in the characterisation of the rescaling-invariant preference relations which rely heavily upon these concepts. In the second part, we describe the standard Support Vector Machine (SVM) method for the classification task. The SVM formulation is used later in Chapter 6 where we consider the effect of rescaling of features in standard SVM. In addition, the explanation lays a foundation on which we can build and develop the idea of maximising margin in the context of preference learning.

Chapter 3: Related Work

In this chapter, we present the background material and a literature review related to preference handling, which is the primary research domain of the thesis. We start with some properties of preference relations and describe different types of pre-orders. Then, we briefly look at the connection between preferences and the decision-making problem. Thereafter, we explore a variety of preference relations that are either (i) based on a utility function such as UTA or the weighted coefficient model, or (ii) more relational and qualitative such as Pareto and lexicographical dominance.

Chapter 4: Learning User Preferences for a Ridesharing Application

This chapter illustrate how a maximum margin preference learning method can be effectively applied to a real-world problem, ride-sharing, to enhance users' satisfaction degrees. We first explain what ride-sharing is, and what are the obstacles for its wide adoption. Proper matching of drivers and riders, as an important challenge in ride-sharing, is described in further detail. We then propose a novel strategy for the matching process that involves learning user preferences. To do so, an SVM-inspired preference learning method is implemented. The experiments show that our method can improve the user's experience in a ride-sharing application.

Chapter 5: Scaling-Invariant Maximum Margin Preference Learning

This chapter includes the central contributions of the thesis. The chapter begins with introducing some notation and then explaining the maximum margin preference relation with respect to that terminology. Afterwards, the core of the chapter is dedicated to characterising preference relations that are not sensitive to the scaling of: (i) preference inputs, (ii) features, and (iii) both inputs and features simultaneously. We then consider how we can deal with inconsistent data. Since the defined preference relations do not generally lead to a total order, we also describe two incremental algorithms to find *optimal* elements, with respect to two different notions of optimality. In the last section, the experimental results draw thorough comparisons between preference relations from different aspects.

Chapter 6: Rescale-Invariant SVM for Binary Classification

Inspired by Chapter 5, we propose a features-rescaling-invariant SVM method for the binary classification task in this chapter. First, we demonstrate that the result of SVM classification could be totally different under a different scaling of features. By making use of some results from the previous chapter, we characterise a method that determines whether the class label that is assigned to an instance, either will vary under different rescaling of features, or is always positive, or is always negative. This method can be seen as a rescaling-invariant classifier that classifies instances into three categories (instead of two), i.e.,

strongly positive, strongly negative, and neutral. We also characterise the situation when the classification of any possible instance by standard SVM is not affected by rescaling of features. Our experiments highlight the value of prediction (i.e., the level of confidence) that is gained by using the proposed method.

1.3 Publications

The work of this thesis has appeared in (or has been submitted for) the following proceedings of journals and conferences, which have been subject to peer-review.

Journal Papers

Submitted: Mojtaba Montazery, and Nic Wilson. Scaling-Invariant Maximum Margin Preference Learning. To Journal of Artificial Intelligence Research (JAIR).

Accepted: Mojtaba Montazery, and Nic Wilson. A New Approach for Learning User Preferences for a Ridesharing Application. In Transactions on Computational Collective Intelligence XXVIII (Due to be published on June 3rd, 2018). Springer.

Conference Papers

Mojtaba Montazery, and Nic Wilson. Dominance and Optimisation Based on Scale-Invariant Maximum Margin Preference Learning. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), pages 1209–1215, 2017.

Mojtaba Montazery, and Nic Wilson. Rescale-Invariant SVM for Binary Classification. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), pages 2501–2507, 2017.

Nic Wilson, and Mojtaba Montazery. Preference Inference Through Rescaling Preference Learning. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), pages 2203–2209, 2016.

Mojtaba Montazery, and Nic Wilson. Learning User Preferences in Matching for Ridesharing. In Proceedings of the Eighth International Conference on Agents and Artificial Intelligence (ICAART-16), volume 2, pages 63–73, 2016.

Chapter 2

Background

2.1 Introduction

This chapter aims to provide some background discussions that the main chapters of the thesis rely on, though we attempt to avoid adding any unnecessary complexity.

The first section includes basic definitions and properties of convex sets that are useful in our work. In the next section, we discuss the conventional Support Vector Machines (SVMs), as a machine learning method, which was first introduced by Vladimir Vapnik and colleagues in [CV95]. SVM has been applied in many real-world problems such as bankruptcy prognosis, face detection, analysis of DNA microarrays, text classification, biological sciences, and breast cancer diagnosis [TK01, CT03, ML05, RW10, BS02, YV13].

2.2 Background I: Convex Sets

As we use the concept of convex sets (and cones) in the thesis quite frequently, this section includes relevant definitions and results.

Definition 1: Convex Set

For some natural number n, a set $\mathcal{C} \subseteq \mathbb{R}^n$ is called *convex* if the line segment between any two points in $\mathcal{C}$ lies entirely in $\mathcal{C}$, i.e., for any $x_1, x_2 \in \mathcal{C}$, and any δ with $0 \leq \delta \leq 1$, we have

$$\delta x_1 + (1 - \delta)x_2 \in \mathcal{C}.$$

Roughly speaking, a set is convex if every point in the set can be seen by every other point, along an unobstructed straight path between them, where unobstructed means lying in the set [BV04, Chapter 2.1].

We call a point of the form $r_1x_1 + \ldots + r_kx_k$ with $r_1, \ldots, r_k \geq 0$ and $r_1 + \ldots + r_k = 1$ a *convex combination* of the points $x_1, \ldots, x_k$. It can be shown that a set is convex if and only if it contains every convex combination of its points.

Definition 2: Convex Hull

The *convex hull* of a set $\mathcal{C} \subseteq \mathbb{R}^n$ is the set of all convex combinations of points in $\mathcal{C}$, i.e.,

$$\{r_1x_1 + \ldots + r_kx_k \,:\, x_i \in \mathcal{C}, r_i \geq 0,\ i = 1, \ldots, k,\ r_1 + \ldots + r_k = 1\}.$$

Definition 3: Convex Cone

A set $\mathcal{C} \subseteq \mathbb{R}^n$ is a

(i) *cone* (or non-negative homogeneous), if for any $x \in \mathcal{C}$ and $r \geq 0$, $rx \in \mathcal{C}$.

(ii) *convex cone* if it is convex and a cone, which implies that for any $x_1, x_2 \in \mathcal{C}$, and any $r_1, r_2 \geq 0$, we have $r_1x_1 + r_2x_2 \in \mathcal{C}$.

Some other definitions refer to the cone with and without $\mathbf{0}$ as the *pointed* and *blunt* cone, respectively [DD11, Ber05]. However, our definition, which is from [BV04], implies that a cone always includes the origin as we only consider cones with origin throughout the thesis.

A point of the form $r_1x_1 + \ldots + r_kx_k$ with $r_1, \ldots, r_k \geq 0$ is called a *non-negative linear combination* (or a *conic combination*) of $x_1, \ldots, x_k$. If $x_1, \ldots, x_k$ are in a convex cone $\mathcal{C}$, then every conic combination of $x_1, \ldots, x_k$ is in $\mathcal{C}$. Conversely, a set $\mathcal{C}$ is a convex cone if and only if it contains all conic combinations of its elements.

Definition 4: Conic Hull

The *conic hull* of a set $\mathcal{C}$, denoted by $co(\mathcal{C})$, is the set of all conic combinations of points in $\mathcal{C}$, i.e.,

$$\{r_1x_1 + \ldots + r_kx_k \,:\, x_i \in \mathcal{C}, r_i \geq 0,\ i = 1, \ldots, k\}.$$

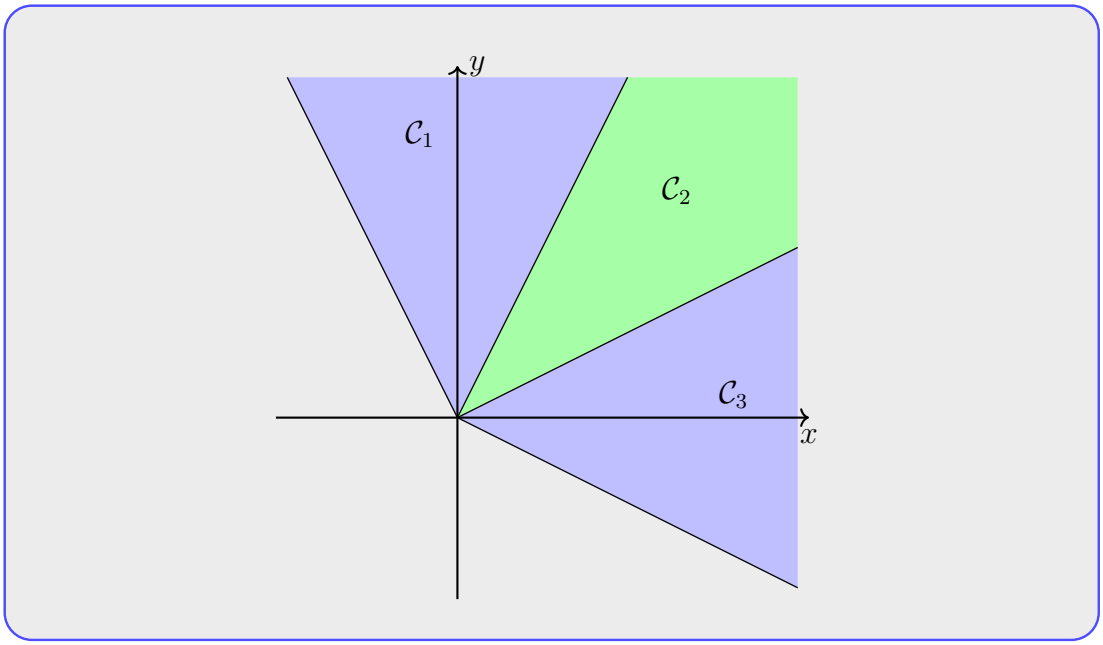

Figure 2.1: The dual cone of $\mathcal{C}_2$ (green) is all the coloured region, i.e., $\mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3$.

The conic hull of $\mathcal{C}$ is in fact the smallest convex cone that contains $\mathcal{C}$ (clearly $co(\mathcal{C}) = \mathcal{C}$ if $\mathcal{C}$ is convex cone itself). We may say that the convex cone $co(\mathcal{C})$ is *generated* by the set $\mathcal{C}$.

Definition 5: Dual Cone

A set $\mathcal{C}^* \subseteq I\!R^n$ is the *dual cone* of a cone $\mathcal{C} \subseteq I\!R^n$, if it is given by

$$\mathcal{C}^* = \{y \in I\!R^n \ : \ x \cdot y \geq 0, \quad \text{for all } x \in \mathcal{C}\},$$

where the dot product $x \cdot y$ is equal to $\sum_{j=1}^{n} x(j)y(j)$, with $x(j)$ and $y(j)$ being the jth component of x and y respectively.

As the name suggests, $\mathcal{C}^*$ is a cone, and is always convex, even when the original cone $\mathcal{C}$ is not. Geometrically, the dual cone $\mathcal{C}^*$ consists solely of all the elements which make an acute angle (i.e., angle less than or equal to $90°$) with every element of the cone $\mathcal{C}$. The definition immediately implies that if $\mathcal{C}_1 \subseteq \mathcal{C}_2$ then $\mathcal{C}_2^* \subseteq \mathcal{C}_1^*$. Figure 2.1 gives an illustration of dual cones.

A polyhedron is defined as the solution set of a finite number of linear equalities and inequalities.

Definition 6: Polyhedron

A set $\mathcal{C} \subseteq \mathbb{R}^n$ is called a *polyhedron*, if there exist $a_j \in \mathbb{R}^n$ and $b_j \in \mathbb{R}$ with $j \in \{1, \ldots, m\}$, such that $x \in \mathcal{C}$ if and only if

$$a_j \cdot x \geq b_j, \quad j \in \{1, \ldots, m\}.$$

A polyhedron is thus the intersection of a finite number of closed half-spaces. It can be easily shown that polyhedra are convex sets [BV04, Chapter 2.2].

2.3 Background II: Support Vector Machines

In the task of classification, a classifier identifies to which of a set of classes a new item belongs, on the basis of a training set whose class membership is known. The basic idea in binary SVM classifier is to find an *optimal* hyperplane which separates the d-dimensional data perfectly into its two classes. Here, "optimal" is used in the sense that the separating hyperplane has the best generalization ability for the unseen data points based on statistical learning theory [LYP10, Vap13]. This optimal separating hyperplane is generated by solving an underlying optimisation problem. Furthermore, the VC-dimension (a measure of a system's likelihood to perform well on unseen data) of SVM can be explicitly calculated, unlike other learning methods like neural networks, for which there is no measure [Bos02].

SVM can also be extended for handling inconsistencies in data (i.e., non-linearly separable data), and solving regression tasks. However, we only discuss the basic linear binary SVM classifier. Overall, SVM is intuitive, theoretically well-founded, and has been shown to be practically successful [Bos02].

We give the formulation of the linear support vector machine for a binary classification task. We define X to be a set of training samples where each training sample is characterised by n input features, and a class label associated with that sample. Features are assumed to be numeric[1].

[1] For ordinal features (e.g., a feature with $\{Cold, Mild, Hot\}$ variables) each value can be replaced by a number, maintaining the order of values. For categorical features, one might use the one-hot encoding (a.k.a. 1-of-k coding scheme) to convert a feature with k cate-

A sample is expressed as a pair of (x, y), where $x \in \mathbb{R}^n$ is the feature vector[2], and $y \in \{+1, -1\}$ indicates the class label for that sample; i.e., $X \subseteq \mathbb{R}^n \times \{+1, -1\}$. The input set can be also represented by two disjoint sets:

- $X^+ = \{x^+ : (x^+, +1) \in X\}$
- $X^- = \{x^- : (x^-, -1) \in X\}$

This means that X^+ (respectively X^-) is the set containing features vectors that is associated with the positive (respectively negative) class label in X. Having assumed that positive and negative samples are linearly separable, we have some separating hyperplane (a line in two-dimensional space) $H = \{\mu : w \cdot \mu + b = 0\}$, for some $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$. Given this separating hyperplane, we have:

$$w \cdot x^+ + b > 0 \qquad \forall x^+ \in X^+ \tag{2.1}$$

$$w \cdot x^- + b < 0 \qquad \forall x^- \in X^- \tag{2.2}$$

or more compactly:

$$y(w \cdot x + b) > 0 \qquad \forall (x, y) \in X. \tag{2.3}$$

Note that a given hyperplane represented by (w, b) can be equally expressed by all pairs (rw, rb) for $r \in \mathbb{R}_+$ ($\mathbb{R}_+$ is the set of strictly positive reals.); i.e., for any $\mu \in H$ and any $r \in \mathbb{R}_+$, $w \cdot \mu + b = 0$ if and only if $rw \cdot \mu + rb = 0$ (for example $2x_1 + 3x_2 + 1 = 0$ and $4x_1 + 6x_2 + 2 = 0$ both represent a same hyperplane). Now, let us choose r given by:

$$r = \frac{1}{\min\limits_{(x,y)\in X} y(w \cdot x + b)}.$$

According to Equation 2.3, this choice of r is strictly positive and thus well-defined. This implies that:

$$\min_{(x,y)\in X} y(rw \cdot x + rb) = 1, \text{ which implies} \tag{2.4}$$

$$y(rw \cdot x + rb) \geq 1 \qquad \forall (x, y) \in X. \tag{2.5}$$

gories to k Boolean features. For example a feature, that represents the type of car with values $\{$*Sedan*, *SUV*, *Hatchback*$\}$, is converted to three binary features: *Is_Sedan*, *Is_SUV*, and *Is_Hatchback*. Clearly, among these three features exactly one of them is true and the two others are false.

[2] $x(k)$ is the score for x regarding the kth feature.

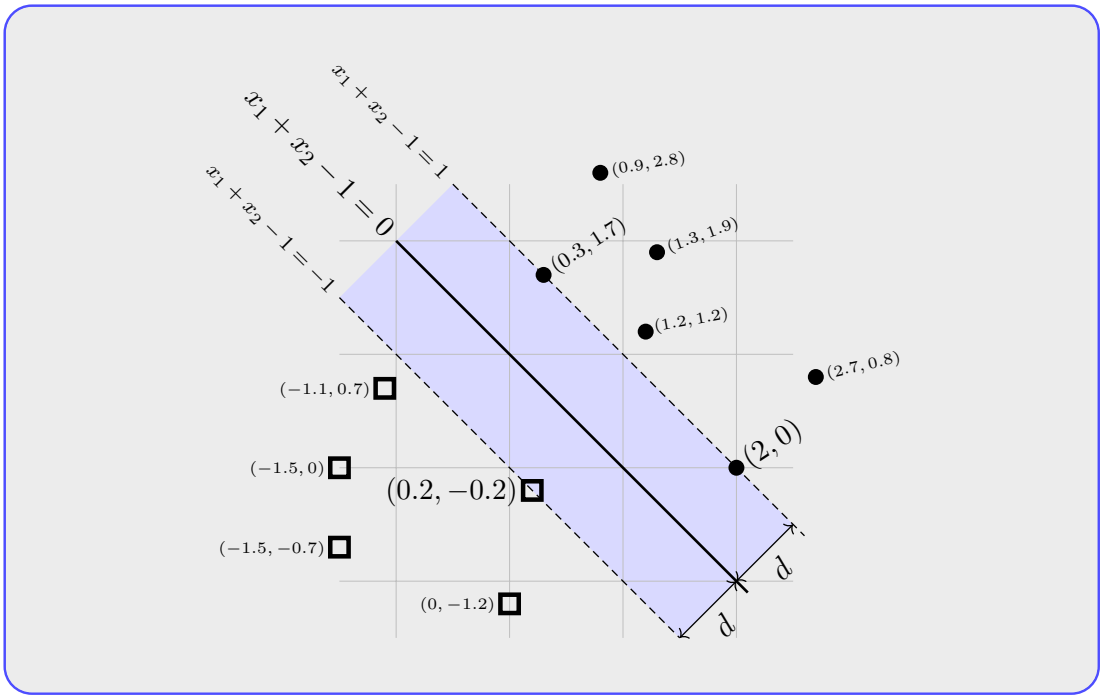

Figure 2.2: The illustration of a set of training samples where bullet points are positive samples and squares represent negative samples. Samples are separated by the hyperplane $x_1 + x_2 - 1 = 0$.

Without loss of generality, we can have a change of variable by using (w, b) instead of (rw, rb) in Equations 2.4 and 2.5. As a result, we say for any separating hyperplane H, there exists $(w, b) \in \mathbb{R}^n \times \mathbb{R}$ such that (w, b) represents H (i.e., $w \cdot \mu + b = 0$ for all $\mu \in H$), and

$$\min_{(x,y)\in X} y(w \cdot x + b) = 1 \tag{2.6a}$$

$$\text{and} \quad y(w \cdot x + b) \geq 1 \qquad \forall (x, y) \in X. \tag{2.6b}$$

Thus, from Equation 2.6a, the closest sample(s) to the separating hyperplane (say x^*) lie(s) on either the hyperplane $w \cdot \mu + b = 1$ if $x^* \in X^+$ or the hyperplane $w \cdot \mu + b = -1$ if $x^* \in X^-$; that means, $|w \cdot x^* + b| = 1$.

Example 1 » Conventional Binary SVM Representation

Consider the black bullets and empty squares in Figure 2.2 are respectively illustrating training samples associated with positive and negative class labels when $n = 2$ (i.e., the number of features is two). The hyperplane $x_1 + x_2 - 1 = 0$ separates data, and the closest points to the hyperplane are

$(0.3, 1.7)$ and $(2, 0)$ from the positive side and the point $(0.2, -0.2)$ from the negative side. As is seen, these positive and negative boundary points lie on two parallel hyperplanes $x_1 + x_2 = 2$ and $x_1 + x_2 = 0$, respectively.

We know that the geometric distance from a hyperplane represented by (w, b) to the point u is

$$\frac{|w \cdot u + b|}{\|w\|},$$

where $\|w\|$ is Euclidean norm of w. Now, define the "*margin*" of a separating hyperplane to be d, where d is the shortest distance from the separating hyperplane to the closest sample x^*. So, we have

$$d = \frac{|w \cdot x^* + b|}{\|w\|} = \frac{1}{\|w\|}$$

Intuitively, SVM looks for the separating hyperplane with the largest margin (i.e., maximising the geometric distance to the closest samples). Hence, SVM searches for a separating hyperplane by maximising $1/\|w\|$ subject to conditions in Equation 2.6. It can be shown that this leads to the following minimisation problem:

$$\begin{aligned} \underset{w}{\text{minimise}} \quad & \frac{1}{2}\|w\|^2 && \text{(2.7a)} \\ \text{subject to} \quad & \\ & y(w \cdot x + b) \geq 1 \qquad \forall (x, y) \in X. && \text{(2.7b)} \end{aligned}$$

We will show that this is a convex *Quadratic Program* (QP); an optimisation problem is called a convex quadratic program if the objective function is convex quadratic, and the constraint functions are affine. That means in a QP, we minimise a quadratic function over a polyhedron [BV04, Section 4.4]. A convex quadratic function for the n-dimensional vector $x \in \mathbb{R}^n$ is formally written as follows:

$$\frac{1}{2}x^T \mathbf{P} x + q \cdot x$$

where x^T is the vector transpose of x, $\mathbf{P}$ is a $n \times n$ symmetric semi-definite matrix, and $q \in \mathbb{R}^n$.

Now in Equation 2.7, clearly, constraints are affine. Also, we can rewrite the

objective function as:

$$\frac{1}{2}\|w\|^2 = \sum_{j=1}^{n} w(j)^2 = \frac{1}{2} w \cdot w = \frac{1}{2} w^T \mathbf{I_n} w$$

where $\mathbf{I_n}$ is the identity matrix of size n. $\mathbf{I_n}$ is obviously symmetric, and also is positive (semi-)definite since for any non-zero w we have $w^T I_n w = \sum_{j=1}^{n} w(j)^2 > 0$. This implies that the objective function here is convex quadratic, and so Equation 2.7 is a QP. Fortunately, many techniques have been developed to solve QP problems [OFG97, Kau98, Van99].

The training samples on the bounding planes, i.e., $w \cdot \mu + b = \pm 1$, are called support vectors. For example in Figure 2.2, $(0.3, 1.7)$ and $(2, 0)$ are positive support vectors, and $(0.2, -0.2)$ is the negative support vector. If we remove any point which is not a support vector from X, the separating hyperplane and subsequently the result of SVM will not change. This is a very nice feature of SVM learning algorithms. Once we have the training result, all we need to keep in our databases are the support vectors.

Assume we have obtained the optimal w from Equation 2.7; we must still determine b to fully specify the hyperplane. To do this, take any support vector (either positive or negative) x^* with the class label y, for which we know that $y(w \cdot x^* + b) = 1$. this gives us:

$$b = y - w \cdot x^*, \tag{2.8}$$

since $y \in \{-1, 1\}$.

2.4 Summary

In this chapter, we provided some introductory material for our work in the thesis. The first section gave some definitions regarding the concept of convex sets, including convex cone and polyhedron. In the next section, we presented the basic form of Support Vector Machines. The formalism that was discussed in that section will be helpful for following the ideas and machinery of the proposed methods in Chapters 4, 5 and 6.

Chapter 3

Related Work

3.1 Introduction

In this chapter, we give a literature review along with some basic definitions related to preference relations and their connection with decision-making problems. In a decision-making problem, the task is to choose a single decision or subset of decisions that are preferred to the other decisions. In this situation, we can have some preference information in different forms associated with decisions. Using this preference information, the set of decisions can be ordered or ranked in relation to each other, and this ordering on the set is called a preference relation.

The outline of this chapter is as follows. In Section 3.2, we give an introduction to preferences and properties of preference relations along with some related definitions. Section 3.3 formalises the concept of preference relation in the context of decision-making problem. We discuss a number of models that represent preference relations in the form of utility functions in Section 3.4. Finally in Section 3.5, some more relational or qualitative forms of preferences representation models are reviewed.

3.2 Preferences

The notion of preference aims to model the choices made by a decision-maker. In AI, an artificial agent acts on behalf of another physical or moral decision-maker. A rational agent, given a set of alternatives, chooses a more attractive alternative over a less desirable, according to the preferences of the user or organization that it acts for. For example in an e-commerce application [BHY96, BHY97], each time the user reviews a product, the system may induce some preferences.

3.2.1 Preference Relation Properties

A preference relation is usually a binary relation in which one element dominates the other one. For some set X, any subset of the Cartesian product of X, i.e., $X^2 = \{(\alpha, \beta) : \alpha, \beta \in X\}$, is a *binary relation* on X. We now define a preference relation that is a type of binary relation.

Definition 7: Preference Relation

A *preference relation* $\succcurlyeq$ on a set X is a binary relation. For any $\alpha, \beta \in X$, if $(\alpha, \beta) \in \succcurlyeq$, then we say α is preferred to β.

This relation is also called an *order relation* as it gives an order (not necessarily a total one) over the set of elements. Given any $\alpha, \beta \in X$, if $(\alpha, \beta) \in \succcurlyeq$ then we can also write $\alpha \succcurlyeq \beta$. Similarly, $\alpha \not\succcurlyeq \beta$ means that α is not preferred to β. For any $\alpha, \beta \in X$ exactly one of the following holds:

(i) $\alpha \succcurlyeq \beta$ and $\beta \succcurlyeq \alpha$.

(ii) $\alpha \succcurlyeq \beta$ and $\beta \not\succcurlyeq \alpha$.

(iii) $\alpha \not\succcurlyeq \beta$ and $\beta \succcurlyeq \alpha$.

(iv) $\alpha \not\succcurlyeq \beta$ and $\beta \not\succcurlyeq \alpha$.

Next, we give some definitions of some basic properties of binary relations.

Remark 8: Binary Relation Properties

A binary relation $\succcurlyeq$ on X is:

(i) reflexive if, $\alpha \succcurlyeq \alpha$ for all $\alpha \in X$;

(ii) irreflexive if, $\alpha \not\succcurlyeq \alpha$ for all $\alpha \in X$;

(iii) symmetric if, $\alpha \succcurlyeq \beta$ then $\beta \succcurlyeq \alpha$ for all $\alpha, \beta \in X$;

(iv) antisymmetric if, $\alpha \succcurlyeq \beta$ and $\beta \succcurlyeq \alpha$ implies $\alpha = \beta$ for all $\alpha, \beta \in X$;

(v) asymmetric if, $\alpha \succcurlyeq \beta$ then $\beta \not\succcurlyeq \alpha$ for all $\alpha, \beta \in X$ (i.e., irreflexive and antisymmetric);

(vi) complete if, $\alpha \succcurlyeq \beta$ or $\beta \succcurlyeq \alpha$ (or both) for all $\alpha, \beta \in X$;

(vii) transitive if, $\alpha \succcurlyeq \beta$ and $\beta \succcurlyeq \gamma$ then $\alpha \succcurlyeq \gamma$ for all $\alpha, \beta, \gamma \in X$.

We distinguish between binary relations based on the properties they hold. We

consider in particular the following four types of order relations.

Definition 9: Order Relations

A binary relation $\succcurlyeq$ on X is a:

(i) preorder on X, if it is reflexive and transitive;

(ii) partial order on X, if it is a preorder and antisymmetric;

(iii) total preorder (a.k.a. weak order) on X, if it is complete and transitive;

(iv) total order on X, if it is total preorder and antisymmetric.

For example, the relation $\geq$ on $\mathbb{R}$ is a total order because it is complete (i.e., $a \geq b$ or $b \geq a$ for all $a, b \in \mathbb{R}$), transitive (i.e., if $a \geq b$ and $b \geq c$ then $a \geq c$ for all $a, b, c \in \mathbb{R}$), and antisymmetric (i.e., if $a \geq b$ and $b \geq a$ then $a = b$ for all $a, b \in \mathbb{R}$).

3.2.2 Associated Relations to a Preorder

For a preorder $\succcurlyeq$ on a set X, we look at some relations that are associated with $\succcurlyeq$. First, we look at the strict or asymmetric part of $\succcurlyeq$, which is defined as follows.

Definition 10: Strict Relation

For a preorder $\succcurlyeq$ on a set X, and any $\alpha, \beta \in X$, the associated *strict* preorder relation $\succ$ is given by:

$\alpha \succ \beta$ if and only if, $\alpha \succcurlyeq \beta$ and $\beta \not\succcurlyeq \alpha$.

This relation $\succ$, is irreflexive and transitive, and represents the notion of strict preference; i.e., for some $\alpha, \beta \in X$, if $\alpha \succ \beta$, then α is strictly preferred to β. For instance, the relation $>$ on $\mathbb{R}$ is the strict part of $\geq$ on $\mathbb{R}$ because it is clearly

irreflexive (i.e., $a \not\succcurlyeq a$ for all $a \in \mathbb{R}$). We now look at the symmetric part of $\succcurlyeq$, which is defined as follows.

Definition 11: Equivalence Relation

For a preorder $\succcurlyeq$ on a set X, and any $\alpha, \beta \in X$, the associated *equivalence relation* $\equiv$ is given by:

$$\alpha \equiv \beta \text{ if and only if, } \alpha \succcurlyeq \beta \text{ and } \beta \succcurlyeq \alpha.$$

This relation $\equiv$ is reflexive, symmetric, and transitive; it represents the notion of equally preferred, if $\alpha \equiv \beta$, then α is equally preferred to β.

This also gives us the notion of an equivalence class; an equivalence relation on X partitions X into some disjoint subsets, called equivalence classes; i.e., any two elements of X are in the same class if and only if they are equivalent. Formally, an equivalence class of X with respect to $\succcurlyeq$ is defined as follows:

Definition 12: Equivalence Class

For a preorder relation $\succcurlyeq$ on a set X, the *equivalence class* of an element $\alpha \in X$, denoted by $[\alpha]_{\succcurlyeq}$, is defined as:

$$[\alpha]_{\succcurlyeq} = \{\beta \,:\, \beta \in X, \beta \equiv \alpha\},$$

where $\equiv$ is the equivalence relation associated with $\succcurlyeq$ as defined in Definition 11.

For any $\alpha, \beta \in X$, and any preorder relation $\succcurlyeq$ on X, $[\alpha]_{\succcurlyeq} = [\beta]_{\succcurlyeq}$ if and only if $\alpha \equiv \beta$. As is well-known, two equivalence classes are either identical or disjoint.

Now, if elements cannot be compared with respect to $\succcurlyeq$, we have the following relation.

Definition 13: Incomparability Relation

For a preorder $\succcurlyeq$ on a set X, and any $\alpha, \beta \in X$, the associated *incomparability relation* (or indifference relation) $\sim$ is given by:

$\alpha \sim \beta$ if and only if $\alpha \not\succcurlyeq \beta$ and $\beta \not\succcurlyeq \alpha$.

For any $\alpha, \beta \in X$, the relation $\alpha \sim \beta$ represents that an individual is uncertain about her preferences between α and β.

3.2.3 Extending Relations

To aid the later comparisons of different preference relations, we define the notion of an extension to a relation [FRW10].

Definition 14: Relation Extension

An extension $\succcurlyeq'$ to a relation $\succcurlyeq$ on a set X, is a binary relation on X such that

for all $\alpha, \beta \in X,\ \alpha \succcurlyeq \beta \Rightarrow \alpha \succcurlyeq' \beta$.

So, if α is preferred to β regarding $\succcurlyeq$, then it is still preferred according to $\succcurlyeq'$. Looking at relations $\succcurlyeq$ and $\succcurlyeq'$ as sets of ordered pairs, we have that $\succcurlyeq \subseteq \succcurlyeq'$.

3.3 Preferences and Decision Making

In this section, we look at preference relations in the context of a general decision-making problem, where we have a set X of decisions, alternatives or choices, and depending on the situation, the task is to choose a single decision or a subset of decisions from this set, given some preference information relating to the decisions . Firstly, we consider a general decision-making problem, which is not specific to any particular decision-making field or area, where the

purpose of this definition is to facilitate the discussion of different preference relations [OW13].

Definition 15: Multi-criteria Decision Problem

A *multi-criteria decision problem* is a tuple $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : j \in \mathcal{S}\}, \{\succcurlyeq_j : j \in \mathcal{S}\}\rangle$, where:

- X is a finite set of decisions, alternatives or choices;
- $\mathcal{S} = \{1, \ldots, n\}$ is a finite set of decision criteria (also attributes, aspects or objectives), where each $j \in \mathcal{S}$ labels some preferential aspect of the problem;
- Each $\alpha \in X$ is represented as a vector of n components with $\alpha(j) \in D_j$ for each $j \in \mathcal{S}$;
- $\succcurlyeq_j$ is a total order on D_j for each $j \in \mathcal{S}$.

In the definition above, $\alpha(j) \in D_j$ denotes the value of alternative $\alpha \in X$ in criterion $j \in \mathcal{S}$. Here, we consider the polarity of domains, where D_j could represent positive preferences or different levels of positive outcomes such as utilities or degrees of satisfaction, or D_j could represent negative preferences, where the values represent different levels of negative outcomes such as costs or degrees of violation. The polarity of the domains determines whether or not larger or smaller values are preferred according to some preference relation; for positive preferences, larger values are preferred, and for negative preferences, smaller values are preferred.

Another consideration in these problems is that we can also have different types of domains. For example, D_j can be *quantitative*, where the difference between two preference degrees has some meaning (interval scales [Kir08]), or if D_j is purely *qualitative* (ordinal scales), then we just have an ordering between preference degrees.

As stated, the order of components of vector $\alpha \in X$ relates to the order of criteria $1, \ldots, n$. This representation is important when the ordering of the preference values (components) needs to be maintained [KRR79]. However, there are certain situations that the comparison of two decisions does not rely

on maintaining the criterion ordering (the ordering of the preference values is not important); for example, in social welfare theory [Sen70], where in a social welfare distribution there is no ordering over the individuals. In such a situation, all criteria have a single domain with a single total order relation over that domain; i.e., for all $j, k \in \mathcal{S}$, $D_j = D_k$ and $\succcurlyeq_j = \succcurlyeq_k$. So, the multi-criteria decision problem for this particular situation can be written in this form $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = D\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \succcurlyeq\}\rangle$. We now define the sorted preference vector with respect to this special case of the multi-criteria decision problem.

Definition 16: Sorted Preference Vector

Given a special case of multi-criteria decision problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = D\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \succcurlyeq\}\rangle$, consider any $\alpha \in X$, and any permutation function $\sigma : \mathcal{S} \to \mathcal{S}$ with respect to α such that $\alpha(\sigma(n)) \succcurlyeq \alpha(\sigma(n-1)) \succcurlyeq \ldots \succcurlyeq \alpha(\sigma(1))$. Then the *sorted preference vector* $\alpha^{\uparrow}$ of α is defined as follows:

for all $j \in \mathcal{S}$, $\alpha^{\uparrow}(j) = \alpha(\sigma(j))$.

Thus, $\sigma(\cdot)$ is a function that accepts the index of one criterion in $\alpha^{\uparrow}$ and gives the index of that criterion in α.

For instance, if $\alpha = (5, 2, 3)$ and $\succcurlyeq$ is simply defined as $\geq$, then $\alpha^{\uparrow} = (2, 3, 5)$. In this example, $\sigma(1) = 2,\ \sigma(2) = 3$, and $\sigma(3) = 1$. The definition implies that $\alpha^{\uparrow}(n)$ has the greatest preference value for α among criteria.

3.4 Utility-based Preference Representation

Representation of preferences is a quite difficult task in general and it is a major goal of decision analysis [Fis70, KRR79]. Many different formalisms have been proposed and studied to represent preferences. One of the main techniques is to assess the value of preferences vector by a utility (cost) function, such that the preference relation $\succcurlyeq$ is defined as: $\alpha \succcurlyeq \beta$ if and only if the utility associated with α is greater than or equal to the utility associated with β. More formally,

the utility function f maps each alternative to a real number; i.e., $f : X \to \mathbb{R}$. The fundamental assumption of utility [Fis70, KR93, VNM07] states that the numbers $f(\alpha)$ and $f(\beta)$ are assigned to the elements α and β in X in such a way that

$$\alpha \succcurlyeq \beta \iff f(\alpha) \geq f(\beta).$$

The advantage of representing preferences in this way is that it defines a total preorder on the set of alternatives, and so it is always possible to answer very common questions such as "which of the two alternatives is better?" or "what is the best choice?". However, the main disadvantage with this form of representation is that it is time-consuming and tedious when one has to deal with the large number of choices with multiple criteria [DHKP11]. Also, the inherent assumption of this representation suggests that all alternatives are comparable, while this might not be necessary the case in practice. If X is finite or countably large then the existence of such utility function f is guaranteed by a certain number of axioms [VNM07].

In this section, we first include some well-known forms of utility function representations, and in 3.4.6, we discuss some methods for learning the utility function mostly based on machine learning techniques.

3.4.1 Additive Independence

A structural assumption of preferences that is the most common approach for evaluating multi-criteria alternatives is to use an additive representation, based on *additive independence* [KRR79, Fis70, KR93, Dye05]. In this additive representation, the utility function is decomposed into sub-utility functions $f_1, \ldots, f_n$—one for each criterion—such that f can be written as:

$$f(\alpha) = \sum_{j \in \mathcal{S}} f_j(\alpha(j)), \tag{3.1}$$

where $\alpha(j) \in D_j$ is the j^{th} component of α (i.e., the value of α in criterion j).

The key condition that is required to have the additive form, is *mutual preference independence* as we define below.

Definition 17: Mutual Preferential Independence

Consider any multi-criteria decision problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : j \in \mathcal{S}\}, \{\succcurlyeq_j : j \in \mathcal{S}\}\rangle$, and any $I \subset \mathcal{S}$. Also, let $\bar{I} = \mathcal{S} - I$, $\mathcal{D}_I = \prod_{j \in I} D_j$ and $\bar{\mathcal{D}}_I = \prod_{j \in \bar{I}} D_j$. Thus, any $x \in \mathcal{D}_I$ is a partial assignment of criteria, as is $\bar{x} \in \bar{\mathcal{D}}_I$, and $(x, \bar{x}) \in \prod_{j \in \mathcal{S}} D_j$ is a complete assignment. Now,

- The set of criteria I is *preferentially independent* of $\bar{I}$ if for all $x, y \in \mathcal{D}_I$, the existence of $\bar{x} \in \bar{\mathcal{D}}_I$ such that $(x, \bar{x}) \succcurlyeq (y, \bar{x})$ implies that for any other $\bar{y} \in \bar{\mathcal{D}}_I$ we also have $(x, \bar{y}) \succcurlyeq (y, \bar{y})$.
- The criteria in $\mathcal{P}$ are *mutually preferentially independent* if for every subset $I \subseteq \mathcal{S}$, the set I is preferentially independent of $\bar{I}$.

When coupled with some technical conditions, mutual preference independence implies the existence of an additive multi-criteria value function for $n \geq 3$ [Dye05]. An example may help to illustrate the idea of preference independence.

Example 2 » Preference Independence

Suppose someone is attempting to evaluate some restaurant meals based on three criteria: cost, the main component of the meal and the side component. We can intuitively say that if two identical meals have different prices (though it does not usually happen in reality), the user always prefers a cheaper one. This means $I = \{cost\}$ is preferentially independent of $\bar{I} = \{main\ component, side\ component\}$. Assume now that the user prefers salad as the side component for fish, and chips for chicken, so for example we have such statements from him saying that $(€10, chicken, chips) \succcurlyeq (€10, chicken, salad)$, but $(€15, fish, salad) \succcurlyeq (€15, fish, chips)$. That implies that $I = \{side\ component\}$ is not preferentially independent of $\bar{I} = \{cost, main\ component\}$, and thus we do not have the mutual preferential independence condition in this example.

3.4.2 Generalized Additive Independence

Additive independence relations require quite strong assumptions regarding the structure of the preferences. However, it seems that our preferences may satisfy some weaker (than additive) forms of independence. i.e., when there is mutual preference dependence between some subsets of criteria [KRR79]. For example, the only attribute that affects how much sugar I want in my hot drink is the type of drink, not the glass and so on.

A more general form of additive independence that allows us to be as general as we wish, is the *generalized additive independence* (GAI) introduced by [Fis67]. The model has gained popularity in the literature because of its additional flexibility [BG95, BBB01, BPPS03, BPPS05, BB07, DGP09].

In the GAI model, independence holds among certain subsets of criteria rather than individual criteria. Formally, consider having k subsets $I_1, \ldots, I_k \subseteq \mathcal{S}$, where $\bigcup_{i=1}^{k} I_i = \mathcal{S}$ and each I_i is preferentially independent from its complement (i.e., $\bar{I}_i$). Also, let the components of the alternative α that are included in the subset I_i, be a partial tuple $\alpha_{[I_i]} = \langle \alpha(j) : j \in I_i \rangle$. Now, the generalized additive independence condition holds if and only if there exists a function of the following form representing the preference relation [BB05, Theorem 1]:

$$f(\alpha) = \sum_{i=1}^{k} f_i(\alpha_{[I_i]}). \tag{3.2}$$

By allowing for subsets containing more than one criterion, we enable capturing preferentially-dependent criteria. Also, note that the subsets are not necessarily disjoint. As a result, we might substantially reduce subset sizes by enabling different subsets to be influenced by the same criterion, without having to combine their associated criteria into a single, large subset.

Example 3 » Generalized Additive Independence

Consider Example 2 where $\mathcal{S} = \{$*cost*, *main component*, *side component*$\}$. *According to the assumptions made there, we have two preferentially independent subsets,* $I_1 = \{$*cost*$\}$ *and* $I_2 = \{$*main component*, *side component*$\}$. *So, for an alternative* α*, the value function will be:*

$$f(\alpha) = f_1(\alpha(\textit{price})) + f_2(\alpha(\textit{main component}), \alpha(\textit{side component})).$$

It should be noted that, unlike the additive independence model, GAI is fully flexible; if we choose $k = 1$ and $I_1 = \mathcal{S}$, we can represent any value function. In the other extreme, if we choose $k = n$ and each subset contains only one criterion, we obtain the additive independence representation.

The GAI representation can be compiled into graphical structures, called GAI networks [GP04]. A GAI network is an annotated graph whose nodes correspond to the criteria in $\mathcal{S}$, and an edge connects the nodes corresponding to a pair of criteria that occur jointly in some subset I_i. GAI networks can be exploited to perform classical optimisation tasks (e.g. finding an alternative with maximal utility) [GPQ08, GPD11].

3.4.3 UTA Representation

The UTA (*UTilités Additives*) method proposed in [JLS82] aims at inducing additive value functions from a given ranking on set of decisions [SGM05]. In this model it is assumed that the domains of criteria are real numbers. The criteria aggregation model in UTA is in the following form:

$$f(\alpha) = \sum_{j \in \mathcal{S}} w_j g_j(\alpha(j)) \tag{3.3a}$$

subject to

$$\sum_{j \in \mathcal{S}} w_j = 1, \tag{3.3b}$$

where $w_j \in \mathbb{R}$ is the weight of g_j, and g_j with $j \in \mathcal{S}$, are non-decreasing real valued functions, named marginal value functions, which are normalized between 0 and 1. This representation is in fact equivalent to Equation 3.1 when the sub-utility function f_j is rewritten as $f_j(\alpha(j)) = w_j g_j(\alpha(j))$.

The UTA method assumes the marginal value functions (i.e., g_j) to be piecewise linear. Then, a special linear programming method is utilised to find some global utility functions so that the rankings obtained through these functions are as consistent as possible with the given ranking.

A variation of UTA that was introduced in [SY85] is UTASTAR. In UTASTAR method, a double positive error function is introduced to be minimised. So,

Equation 3.3a becomes:

$$f(\alpha) = \sum_{j \in \mathcal{S}} w_j g_j(\alpha(j)) - \sigma^+(\alpha) + \sigma^-(\alpha), \tag{3.4}$$

where σ^+ and σ^- are the overestimation and the underestimation error respectively.

One of the main assumptions in UTA is that the marginal value functions are monotonic. This assumption, although widely used, is sometimes not applicable to real-world situations. One way to deal with non-monotonic preferences is to divide the range of a criterion into intervals, so that the preferences are monotonic in each interval, and then treat each interval separately [KR93]. Inspired by this technique, [DZ95] presents a variation of UTASTAR for the assessment of non-monotonic marginal value functions.

Another assumption in UTA methods is that the marginal value functions are piecewise linear. The choice of such functions is historically motivated by the opportunity of using linear programming solvers. Although piecewise linear functions are well-suited for approximating monotone continuous functions, their lack of smoothness (differentiability) may make them seem "not natural" in some contexts, especially for economists [SGMP18]. Abrupt changes in slope at the breakpoints are difficult to explain and justify. Because of this, some methods concerning this fact have been developed. The MIIDAS system [SSY99] proposes tools to model marginal value functions; possibly non-linear (and even non-monotone) shapes of marginals can be chosen from parametrised families of curves. In [BKU02], the authors propose an inference method based on linear programming that infers quadratic utility functions in the context of an application to the banking sector. [SGMP18] propose to infer polynomials and splines functions for marginals by making use of semi-definite programming instead of linear programming.

Another extension of UTA methods refers to the intensity of the user's preferences; e.g., the preference of alternative a over b is stronger than the preference of b over c. That includes the model described in [SS73] in which a series of constraints are allowed to be added during the linear programming formulation, or [OK89] where a ratio scale is used in order to express the intensity of preferences, or [BP13] that attaches a likelihood degree to each pair in the given ranking.

Other techniques, named meta-UTA, aim at the improvement of the value func-

tion with respect to near optimality analysis or to its exploitation. These include the UTAMP2 method proposed in [BS01], and [DYZ90].

As opposed to traditional UTA methods, some more recent works that propose a single utility function include [GMS08, FGS09, KGS12].

3.4.4 Weighted Coefficient Model

This is a special case of UTA representation, where $g_j(\alpha(j)) = \alpha(j)$ for all $j \in \mathcal{S}$. Thus, the utility function will be in the following form:

$$f(\alpha) = \sum_{j \in \mathcal{S}} w_j \alpha(j), \tag{3.5}$$

where for all criterion j in $\mathcal{S}$, w_j is a (usually non-negative) number which represents the relevance importance of that criterion.

There are many studies attempting to systematically assign weights to criteria with respect to perceptions of the user. As one of the very first instances, [Eck65] provides a method for choosing weights for each of the six criteria involved in designing a specific air defence and a general air defence systems, and selecting a personnel subsystem manager for a development program.

Another instance is the AHP (Analytic Hierarchy Process) method that was developed in [Saa88, Saa03, Saa05]. In this method, the user is asked to state quantitatively the relative importance of the criterion j over the criterion k. The answer is interpreted as being an estimate of w_j/w_k. Having n criteria, we have $\frac{n(n-1)}{2}$ pairs of criteria in which the relative importance must be determined. However, to do so, it can be seen that only $n-1$ pairs of carefully chosen comparisons are sufficient, because for example, knowing the values of w_1/w_2 and w_2/w_3 gives us w_1/w_3. After acquiring these values for all pairs of criteria, a symmetric matrix $\mathbf{A_{n \times n}}$ is constructed such that $a_{jk} = w_j/w_k$. This matrix is called the *judgement matrix*. Assuming the vector w with n components consists of weights, it can be seen that $\mathbf{A}w$ will be equal to nw, and:

$$\mathbf{A}w = nw \Rightarrow (\mathbf{A} - n\mathbf{I_n})w = \mathbf{0}, \tag{3.6}$$

where $\mathbf{I_n}$ is the identity matrix of size n. This is a system of homogeneous linear equations, and it is shown that for this structure of matrix it has a non-trivial solution for the vector w.

Another method for choosing weights, SVMRank [Joa02], selects a weight vector that maximises the margin (or equivalently has the minimum Euclidean norm). We will discuss this method extensively in Sections 4.4 and 5.2.2 later.

Another approach that has been explored, for example in [Raz14, MRW13], considers the weight vectors that are compatible with input data. That is, the weight vector w is compatible if and only if $w \cdot a \geq w \cdot b$, for all input comparisons $a \succcurlyeq b$. Then, α is preferred to β with respect to this relation if for all compatible w, $w \cdot \alpha \geq w \cdot \beta$. We explain this approach further in Section 5.2.1.

3.4.5 Choquet Integral Method

The idea of using the Choquet integral [Cho54] for preferences representation was introduced in [Sch86]. The Choquet integral is an evaluation function that performs a weighted aggregation of criterion values using a capacity function assigning a weight to any coalition of criteria, thus enabling positive and/or negative interactions among them and covering an important range of possible decision behaviours [BPV17]. The Choquet integral has received much attention recently [DF05, Koj07, BMM08, GL10], and some justifications for the use of Choquet integral can be found in [GL05, GR08]. We now introduce some formal definitions regarding Choquet integral representation.

Definition 18: Capacity Function

Consider the power set of the criteria set $\mathcal{S}$ as $\mathbf{P}(\mathcal{S})$. A *capacity* on $\mathcal{S}$ is a set function $\Omega : \mathbf{P}(\mathcal{S}) \to \mathbb{R}$ satisfying the following conditions:

(i) $\Omega(\emptyset) = 0$,

(ii) for any $I, K \subseteq \mathcal{S}$, $I \subseteq K \implies \Omega(I) \leq \Omega(K)$.

The capacity is normalized if $\Omega(\mathcal{S}) = 1$.

$\Omega(I)$ can be thought of representing the weight attached to the subset I, for any $I \subseteq \mathcal{S}$.

Definition 19: Choquet Integral

Consider the multi-criteria decision problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = I\!R\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$. Recall from Definition 16 that $\alpha^{\uparrow}$ is the sorted preference vector and $\sigma(.)$ is the permutation function for alternative $\alpha \in X$ (i.e., for all $j \in \mathcal{S}$, $\alpha^{\uparrow}(j) = \alpha(\sigma(j))$). The *Choquet integral* is a function $f_\Omega : X \to I\!R$ with respect to a capacity Ω on $\mathcal{S}$ such that:

$$f_\Omega(\alpha) = \sum_{j \in \mathcal{S}} [\Omega(I_{\alpha,j}) - \Omega(I_{\alpha,j+1})]\ \alpha^{\uparrow}(j),$$

where $I_{\alpha,n+1} = \emptyset$ and $I_{\alpha,j} = \{\sigma(j), \ldots, \sigma(n)\} \subseteq \mathcal{S}$.

The following example clarifies this rather complicated definition.

Example 4 » Choquet Integral Function Illustration

Consider a problem defined on 3 *criteria, i.e.* $\mathcal{S} = \{1, 2, 3\}$*, two alternatives* $\alpha = (5, 2, 3)$ *and* $\beta = (1, 6, 4)$*, and the following capacity* Ω *on* $\mathcal{S}$*:*

	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
Ω	0	3	1	6	5	7	6	9

In this setting, we can see that $\alpha^{\uparrow} = (2, 3, 5)$ *and* $\beta^{\uparrow} = (1, 4, 6)$*. Thus,*

- $I_{\alpha,1} = \{1, 2, 3\}$*,* $I_{\alpha,2} = \{1, 3\}$*, and* $I_{\alpha,3} = \{1\}$*;*
- $I_{\beta,1} = \{1, 2, 3\}$*,* $I_{\beta,2} = \{2, 3\}$*, and* $I_{\beta,3} = \{2\}$*.*

Now, the computation of the Choquet value of α *and* β *regarding the capacity* Ω *gives:*

$$\begin{aligned} f_\Omega(\alpha) &= 2\ [\Omega(\{1,2,3\} - \Omega(\{1,3\}))] + 3\ [\Omega(\{1,3\} - \Omega(\{1\}))] + 5\ \Omega(\{1\}) \\ &= 2\ (9-7) + 3\ (7-3) + 15 \\ &= 4 + 12 + 15 = 31 \end{aligned}$$

and

$$\begin{aligned} f_\Omega(\beta) &= 1\ [\Omega(\{1,2,3\} - \Omega(\{2,3\}))] + 4\ [\Omega(\{2,3\} - \Omega(\{2\}))] + 6\ \Omega(\{2\}) \\ &= (9-6) + 4\ (6-1) + 6 \\ &= 3 + 20 + 6 = 29. \end{aligned}$$

Since $f_\Omega(\alpha) > f_\Omega(\beta)$, we say that α is strictly preferred to β.

In order to use this method, prior identification of the capacity function is required. A review of approaches for capacity determination is given in [GKM08].

If the capacity is additive, i.e., $\Omega(I \cup K) = \Omega(I) + \Omega(K)$ for all disjoint subsets $I, K \subseteq \mathcal{S}$, it can be seen that the Choquet integral will reduce to:

$$f_\Omega(\alpha) = \sum_{j \in \mathcal{S}} \Omega(\{\sigma(j)\})\ \alpha^{\uparrow}(j) = \sum_{k \in \mathcal{S}} \Omega(\{k\})\ \alpha(k).$$

This form is basically a weighted coefficient model, meaning that the weighted coefficient model is a special case of Choquet integral.

3.4.6 Learning a Utility Function

Here, we discuss learning the utility function in particular for the purpose of ranking alternatives. Ranking problems arise quite naturally in many application areas. One interesting problem is learning to rank possible recommendations for new products [BHK98]. Also, ranking functions are at the core of search engines and they directly influence the relevance of the search results and users' search experience (see e.g., works by Joachims and his colleagues [Joa02, KT03, JGP+05, RJ05]). The trend for ranking alternatives continues to this day and several methods have been proposed incorporating the advances in machine learning such as SVM and gradient boosting.

Authors in [HGO00] cast the problem of learning to rank as ordinal regression, that is, learning the mapping of an input vector to a member of an ordered set of numerical ranks. They model ranks as intervals on the real line, and consider loss functions that depend on pairs of examples and their target ranks. The positions of the rank boundaries play a critical role in the final ranking function.

A group of researchers from Microsoft Research developed RankNet [BSR+05] which does not need finding rank boundaries. The method proposed an optimisation approach using an objective function based on Bradley-Terry models for paired comparisons, and explored neural networks (gradient descent) for learning the ranking function. In [Tes89], Tesauro proposed a symmetric neural network architecture that can be trained with representations of two states and a training signal that indicates which of the two states is preferable. The elegance of this approach comes from the property that one can replace the two symmetric components of the network with a single network, which can subsequently provide a real-valued evaluation of single states. Other methods making use of neural networks include RankProp [CBM96] and [HHR+03]. TrueSkill [HMG07] is a Bayesian model that can be applied for learning to rank.

Based on boosting approach in machine learning [Sch03], RankBoost [FISS03] approximates the target ranking by combining many weak rankings (i.e., ranking with ties) from the given preferences. However, the choice for selecting weak learners is quite limited and is less flexible when dealing with the complicated features. Authors in [ZCSZ07] attempts to address this limitation by proposing a learning framework for preference data by using regression. The study transforms the problem of learning ranking functions in terms of a sequence of problems of learning regression functions. It demonstrates, in particular, the application of the gradient descent regression methodology to the objective function.

SVMRank [Joa02]—inspired by Support Vector Machines—is applied for web search ranking by using click-through data, i.e., the query-log of the search engine in connection with the log of links the users clicked on. In the study, the proposed method is trained with a couple of hundred examples, and showed better ranking results in comparison with the Google search engine (of course in the time of publication of the paper in 2002). As we previously mentioned, we will discuss this method in further details later in Sections 4.4 and 5.2.2. Another method based on SVM is SVMCompare [HSS14]. It differs from SVMRank in that it also considers the case where both alternatives in a pairwise comparison are judged to be equal.

3.5 Relational Preference Models

So far, we have seen several well-known representations of preferences in the form of a utility function, along with some methods to learn the utility function mainly focused on the task of ranking alternatives. In this section, we review some other forms of preference representation models that are not (at least directly) using the concept of the utility function, but rather exploiting more relational or qualitative forms of preferences representation.

One well-known relational representation of preferences is the CP-net [BBHP99]. CP-nets can deal with a more general form of qualitative statements, based on a *ceteris paribus* semantics ("all else being equal"). With a ceteris paribus interpretation, the preference relation depends only on features that are different; as an instance of this semantic, we can have "I prefer a chicken meal to a fish meal, provided all other properties are the same". Users may find it cognitively easier to express their preferences with such statements, and as a result, more compact communication of the preference model with the user is enabled [DHKP11]. From these preference statements, a graphical structure for the CP-net is induced. This graph leads in general only to a partial order over alternatives; therefore, a CP-net may not order sufficient alternatives to be useful in practice [Wal07]. However, with respect to a given CP-net graph, we can answer some common queries such as finding the optimal alternative, comparing two particular alternatives and so on; for example, if the graph is acyclic then preference statements are consistent [BBD$^+$04, Theorem 1], and in that case, the nodes can be ordered topologically which leads to finding the optimal alternative in the linear time [BBD$^+$04, Corollary 4]. Determining whether one alternative is more preferred than another varies depending on the structure of the network from polynomial to PSPACE-complete [GLTW08]. To reduce this complexity, various approximations have been suggested [DPR$^+$06, DRVW09]. A number of extensions for CP-nets have been proposed including mCP-nets [RVW04] to represent the preferences of multiple agents (each agent has its own CP-net and these are combined using voting rules), TCP-nets [BDS06] to represent trade-offs (for example, "price" is more important to me than "side component"), and [Wil04] that can represent stronger conditional preference statements.

Another preference relation is proposed in [CSS99] by developing an online algorithm to learn a binary preference predicate $P(\alpha, \beta)$, which predicts whether

α is preferred to β or vice versa. This algorithm is based on the "Hedg" algorithm [FS95], and has two phases. In the first phase, this predicate is trained on the basis of exemplary preferences in the form of pairwise comparisons. Afterwards, a final ordering is found in a second phase by deriving (an approximation of) a ranking that is maximally consistent with these predictions.

In [GMPU06, UPGM09], a simple algorithm is proposed to find a weak order with ties, called a *bucket order* (where each bucket corresponds to an equivalence class), from a set of pairwise comparisons. The method generates the *pair order matrix* C. The entry C_{tu} indicates how many users preferred item t to item u. The pair order matrix is normalized so that $C_{tu} + C_{ut} = 1$ for all t and u. If for example only one user is involved and his/her preferences are consistent, then consequently, the elements of the matrix are restricted to $\{0, 1/2, 1\}$, because either the user has preferred t to u, or u to t, or u and t are indifferent for the user. In the core of algorithm, u and t are assigned to a same bucket if $\frac{1}{2} - \beta \leq C_{tu} \leq \frac{1}{2} + \beta$, where $\beta \in [0, 1]$ is a parameter that is set to $1/4$ by default. If $C_{tu} < \frac{1}{2} - \beta$, then the item t is included in the subset L_u (left side of u), and similarly if $C_{tu} > \frac{1}{2} + \beta$, then the item t is included in the subset R_u. The algorithm will run iteratively on generated subsets (i.e., L_u and R_u), and finally, a total order of "buckets" is returned. The authors have shown that while computing the optimal bucket order from matrix C is NP-hard, the expected running time for the proposed algorithm is $O(n \log n)$. Furthermore, the algorithm has a bounded approximation ratio; it is 9 times less accurate (in the sense defined in the paper) than the algorithm that finds the optimal ordering, and if C satisfies also the triangle inequality the algorithm is 5 times worse. For the restricted version (i.e., values in $\{0, 1/2, 1\}$), the bounds ratio without and with triangle inequality, are 5 and 3, respectively.

We now define a series of preference dominance relations in separate sections.

3.5.1 Pareto Dominance

The (weak) Pareto dominance relation [Par71] is a partial order relation that prefers decisions that are at least as good in every criterion (in the strict version the preference is strictly better in at least one criterion [Sen70, Chapter 2]).

Definition 20: Weak Pareto Dominance

Consider any multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : j \in \mathcal{S}\}, \{\succcurlyeq_j : j \in \mathcal{S}\}\rangle$, and any $\alpha, \beta \in X$. Then, α weakly *Pareto dominates* β with respect to $\mathcal{P}$ if and only if

$$\text{for all } j \in \mathcal{S},\ \alpha(j) \succcurlyeq_j \beta(j).$$

In some situations where

(i) the domains of criteria are the same (i.e., for all $j, k \in \mathcal{S}$, $D_j = D_k$);

(ii) the ordering defined on each domain is the same (i.e., for all $j, k \in \mathcal{S}$, $\succcurlyeq_j$ equals $\succcurlyeq_k$); and

(iii) the order of decision criteria is not important (i.e., we can conceptually compare the value of e.g., criterion j with the value of criterion k),

then we can also look at sorted preference vector of each alternative (see Definition 16), and define Sorted Pareto dominance (or Ordered/Symmetric Pareto [KP08, DPT13]) accordingly.

Definition 21: Weak Sorted Pareto Dominance

Consider any special case of multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = D\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \succcurlyeq\}\rangle$, and any $\alpha, \beta \in X$. Then, α weakly *Sorted Pareto dominates* β if and only if

$$\text{for all } j \in \mathcal{S},\ \alpha^{\uparrow}(j) \succcurlyeq \beta^{\uparrow}(j).$$

It can be shown that sorted Pareto dominance extends Pareto dominance relation [OW12, Proposition 2]; i.e., if α weakly Pareto dominates β then α also weakly sorted Pareto dominates β.

Example 5 » Pareto Dominance Illustration

Consider a problem defined on 3 *criteria, i.e.* $\mathcal{S} = \{1, 2, 3\}$*, and three alternatives* $X\{\alpha = (3, 5, 2), \beta = (6, 1, 2), \gamma = (1, 5, 3)\}$.

In this setting, we can see that $\alpha^{\uparrow} = (2, 3, 5)$*,* $\beta^{\uparrow} = (1, 2, 6)$ *and* $\gamma^{\uparrow} = (1, 3, 5)$*. Clearly, no element in* X *weakly Pareto dominates any the other alternative. However,* α *sorted Pareto dominates* γ*.*

3.5.2 Maximin Dominance

The Maximin relation, which is a total pre-order, is also defined for the special case of the multi-criteria decision problem as explained above. When comparing any two alternatives, the (weak) Maximin relation [Raw09, Wal50] prefers the alternative that is as good as the other alternative in the minimum criteria value. Minimax [VN59] is the counterpart of this relation when criteria are in the form of cost rather than utility.

Definition 22: Weak Maximin Dominance

Consider any special case of multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = D\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \succcurlyeq\}\rangle$, and any $\alpha, \beta \in X$. Then, α weakly *Maximin dominates* β if and only if

$$\min_{j \in \mathcal{S}} \alpha(j) \succcurlyeq \min_{j \in \mathcal{S}} \beta(j).$$

It can be said that α weakly Maximin dominates β if and only if $\alpha^{\uparrow}(1) \succcurlyeq \beta^{\uparrow}(1)$, because the first component in the sorted vector of a preference alternative has the minimum criteria value in that alternative. The weak Maximin dominance extends weak sorted Pareto dominance relation (see e.g., [O'M13, Chapter 2] for proof).

In Example 5, it can be seen that α maximin dominates β and γ, and since $\beta^{\uparrow}(1) = \gamma^{\uparrow}(1) = 1$, β and γ weakly maximin dominates each other.

3.5.3 Lexicographical Dominance

Lexicographic preference models [Fis74, FHWW10, GRW15] are one of the simplest but most intuitive preference representations. The model, which leads to a total order on alternative, is based on a pre-assumption that an order of importance on the criteria set is predefined. Then, the superiority of an alternative to another one in the criterion j, vetoes the effect of all the criteria that are less important than the criterion j. That means for example if an alternative has a better value than another alternative on the most important criterion, then it is considered to be better overall, regardless of how poor the values of the rest of criteria are.

Although that assumption in the lexicographical model may sound too restrictive, [YWL+11] argues that the model is still plausible because several studies on human decision making [CS99, FSS+89, WK94] experimentally demonstrate that humans often make decisions using lexicographic reasoning instead of mathematically more sophisticated methods such as linear additive value maximization [Daw79].

Definition 23: Weak Lexicographical Dominance

Consider any multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : j \in \mathcal{S}\}, \{\succcurlyeq_j : j \in \mathcal{S}\}\rangle$, and any $\alpha, \beta \in X$. Assume that the criteria set is ordered from the least to the most important criterion, so that the first criterion is the least important one. Then, α weakly *Lexicographically dominates* β if and only if either $\alpha = \beta$ or there exists some $j \in \mathcal{S}$ such that

(i) for all $k \in \{j+1, \ldots, n\}$, $\alpha(k) \equiv_k \beta(k)$ (i.e., $\alpha(k) \succcurlyeq_k \beta(k)$ and $\beta(k) \succcurlyeq_k \alpha(k)$); and

(ii) $\alpha(j) \succ_j \beta(j)$.

In Example 5, γ lexicographically dominates α and β, because $\gamma(3) = 3 > \alpha(3) = \beta(3) = 2$. For α and β, as the most important component has the same value, we check the next component where $\alpha(2) = 5 > \beta(2) = 1$; that implies α lexicographically dominates β.

Again we can define a version of this model, called Leximin [BJ88, Far93], for

the case when the ordering of criteria is not important by making use of the sorted vector. In this case, the most important criterion of an alternative is considered the one that has the maximum value among all criteria. Leximax [Ehr99] is the counterpart of this relation when all criteria are in the form of cost rather than utility.

Definition 24: Weak Leximin Dominance

Consider any special case of multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = D\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \succcurlyeq\}\rangle$, and any $\alpha, \beta \in X$. α weakly *Leximin dominates* β if and only if either $\alpha = \beta$ or there exists some $j \in \mathcal{S}$ such that

(i) for all $k \in \{j+1, \ldots, n\}$, $\alpha^{\uparrow}(k) \equiv_k \beta^{\uparrow}(k)$; and

(ii) $\alpha^{\uparrow}(j) \succ_j \beta^{\uparrow}(j)$.

The weak Leximin dominance extends the weak Maximin dominance relation (see e.g., [O'M13, Chapter 2] for proof).

In Example 5, $\beta^{\uparrow}(3) = 6 > \alpha^{\uparrow}(3) = 5$ means that β Leximin dominates α. Then, α Leximin dominates γ because $\alpha^{\uparrow}(3) = \gamma^{\uparrow}(3) = 5$, $\alpha^{\uparrow}(2) = \gamma^{\uparrow}(2) = 3$, and $\alpha^{\uparrow}(1) = 2 > \gamma^{\uparrow}(1) = 1$.

3.5.4 Minimax Regret Dominance

The Minimax Regret [LS82, BPPS05, BB07] relation, which is a total pre-order, looks to minimise the worst-case regret (loss), where the regret of an alternative with respect to another one is the difference between the maximum preference values; the maximum regret of an alternative is the maximum regret over all decisions.

This relation is defined for another special case of the multi-criteria problem, $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, when the domain of all criteria is the real numbers. The reason is that we want to aggregate the values of different criteria. The following definition quantifies the regret caused by choosing the alternative α instead of β.

Definition 25: Regret of α with respect to β

Consider the multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, and any $\alpha, \beta \in X$. The regret of α with respect to β, denoted by $R(\alpha, \beta)$ is

$$R(\alpha, \beta) = \max_{j \in \mathcal{S}}(\beta(j) - \alpha(j)).$$

Next, the notion of maximum regret of choosing an alternative is given.

Definition 26: Maximum regret of an alternative

Consider the multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, and any $\alpha \in X$. The maximum regret of α, denoted by $MR(\alpha, X)$ is

$$MR(\alpha, X) = \max_{\beta \in X} R(\alpha, \beta).$$

Now, we can define Minimax regret dominance that is based on the maximum regret of two alternatives.

Definition 27: Weak Minimax Regret Dominance

Consider the multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, and any $\alpha, \beta \in X$. α weakly *Minimax regret dominates* β if and only if

$$MR(\alpha, X) \leq MR(\beta, X).$$

Example 6 » Minimax Regret Dominance Illustration

Consider the setting of the problem defined in Example 5, where $X = \{\alpha = (3, 5, 2), \beta = (6, 1, 2), \gamma = (1, 5, 3)\}$. *Then, we have:*

- $R(\alpha, \beta) = 3$, $R(\alpha, \gamma) = 1 \implies MR(\alpha, X) = 3$;
- $R(\beta, \alpha) = 4$, $R(\beta, \gamma) = 4 \implies MR(\beta, X) = 4$; *and*
- $R(\gamma, \alpha) = 2$, $R(\gamma, \beta) = 5 \implies MR(\gamma, X) = 5$.

This implies that α *minimax regret dominates* β, *and* β *minimax regret dominates* γ.

3.5.5 Generalised Lorenz Dominance

The Generalised Lorenz Dominance relation [Atk70, Sho83], which is a total order, is a refinement of Pareto dominance, and used in fair optimization problems when fairness refers to the idea of favouring Pareto-dominant solutions having a well-balanced utility profile [GPD11]. Again, for this relation, we need to have real numbers for criteria values.

Definition 28: Generalized Lorenz Curve of α

Consider the multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, any $\alpha \in X$, and any $j \in \mathcal{S}$. The *Generalized Lorenz Curve* of α for the component j, denoted by $LR(\alpha, j)$ is

$$LR(\alpha, j) = \sum_{k=1}^{j} \alpha^{\uparrow}(k),$$

where $\alpha^{\uparrow}$ is the sorted vector associated with α.

This gives us the Generalized Lorenz dominance relation as follows.

Definition 29: Weak Generalized Lorenz Dominance

Consider the multi-criteria problem $\mathcal{P} = \langle X, \mathcal{S}, \{D_j : \forall j \in \mathcal{S}, D_j = \mathbb{R}\}, \{\succcurlyeq_j : \forall j \in \mathcal{S}, \succcurlyeq_j = \geq\}\rangle$, and any $\alpha, \beta \in X$. α weakly *Generalized Lorenz dominates* β if and only if

for all $j \in \mathcal{S}$, $LR(\alpha, j) \geq LR(\beta, j)$.

Example 7 » Generalized Lorenz Illustration

Consider the setting of the problem defined in Example 5, where $\alpha^{\uparrow} = (2, 3, 5)$, $\beta^{\uparrow} = (1, 2, 6)$ *and* $\gamma^{\uparrow} = (1, 3, 5)$. *Then, we have:*

- $LR(\alpha, 1) = 2$, $LR(\alpha, 2) = 5$, *and* $LR(\alpha, 3) = 10$;
- $LR(\beta, 1) = 1$, $LR(\beta, 2) = 3$, *and* $LR(\beta, 3) = 9$;
- $LR(\gamma, 1) = 1$, $LR(\gamma, 2) = 4$, *and* $LR(\gamma, 3) = 9$.

This implies that α *generalized Lorenz dominates* γ, *and* γ *generalized Lorenz dominates* β.

3.6 Chapter Conclusion

In this chapter, we presented some introductory material for preferences and discussed various properties that a preference relation can take; this forms the background to our preference handling techniques in this thesis.

We discussed different utility-based preference representation methods such as additive independence, GAI, UTA, and Choquet Integral. In particular, we described the weighted coefficients model for preferences, which is closely related to the work on preferences in this thesis.

We also explained some relational preferences representations including Pareto, Maximin, Lexicographical, Minimax Regret, and Generalized Lorenz.

Chapter 4

Learning User Preferences for a Ridesharing Application

4.1 Introduction

Ridesharing (a.k.a. *carpooling* and *lift-sharing*) is a mode of transportation in which individual travellers share a vehicle for a trip. Ridesharing has the potential to relieve some transportational issues such as traffic congestion, pollution and high travel costs. In this chapter, we focus on the process of matching drivers and prospective riders more effectively, which is a crucial challenge in ridesharing. A novel approach is proposed in ride-matching which involves learning user preferences regarding the desirability of a choice of matching; this could then maintain high user satisfaction, thus encouraging repeat usage of the system. An SVM inspired method is developed which is able to learn a utility function from a set of pairwise comparisons, and predicts the satisfaction degree of the user with respect to specific matches. To assess the proposed approach, we conducted some experiments on a commercial ridesharing data set. We compare the proposed approach with five rival strategies and methods, and the results clearly show the merits of our approach for matching drivers and riders.

The rest of this chapter is structured as follows. We give some background about ridesharing along with some opportunities and challenges. Among the challenges, in Section 4.3, we focus on the automated ride-matching problem and explain how user preferences could be considered in the matching process. In Section 4.4, a method to learn user preferences is described in detail. Section 4.5 evaluates the presented approach on a real ridesharing database. Finally, in Section 4.6, we summarise the main remarks and discuss some directions for future research.

4.2 Background

Increasing the number of travellers per vehicle trip by effective usage of spare car seats in ridesharing, may, of course, enhance the efficiency of private transportation, and contribute to reducing traffic congestion, fuel consumption, and pollution. Moreover, ridesharing allows users to split travel costs such as fuel, toll, and parking fees with other individuals who have similar itineraries and time schedules. Conceptually, ridesharing is a system that can combine the flexibility and speed of private cars with the reduced cost of fixed-line systems such as buses or subways [FDO+13, AESW11].

Ridesharing is quite an old concept; it was first used in the USA during World War II to conserve resources for the war. It reappeared as a result of the oil crisis in 1970s which led to the emergence of the first ridesharing algorithms. Nevertheless, ridesharing usage declined drastically between the 1970s and the 2000s due to the decrease in the price of fuel and vehicle ownership cost [CS12].

Furthermore, there are some challenges that have inhibited wide adoption of ridesharing. A few of the most important of those are listed as follows:

Riding with Strangers Surveys suggest that there is little interest in sharing a ride with strangers, because of personal safety concerns. This phenomenon is referred to as *Stranger Danger* and could be alleviated by incorporation of social networks [AAM11, FDO+13]. [CKPQ10] conducted a survey among students of a university which shows that while only 7% of participants would accept rides from a stranger, 98% and 69% would accept rides from a friend and the friend of a friend, respectively.

Reliability of Service One of the largest behavioural challenges is the perception of low reliability in ridesharing arrangements; the parties may not necessarily follow through on the agreed-upon ride. For instance, if the driver has an unexpected appointment or emergency, the passenger may be left with no ridesharing option; or, from the other side, drivers might be required to wait because of a passenger being late [AAM11].

Schedule Flexibility The lack of schedule flexibility has been one of the longest-running challenges in ridesharing arrangements. Drivers and passengers often agree on relatively fixed schedules and meeting locations, not allowing much flexibility. It is interesting to note that increasing the flexibility and increasing the reliability of ridesharing arrangements are often conflicting objectives [AAM11].

Ride Matching Optimally matching riders and drivers—or at least getting a good match—is among the most important challenges to overcome. This can lead to a complicated optimisation problem due to a large number of factors involved in the objective function. We will discuss this aspect of ridesharing further in Section 4.3.

Despite the above barriers to ridesharing, the demand for ridesharing services has increased again sharply in recent years, generating much interest along with media coverage [Sar06]. This boost in ridesharing is mainly associated with a relatively new concept in ridesharing, *dynamic* or *real-time* rideshar-

ing. Dynamic ridesharing refers to a system which supports an automatic ride-matching process between participants at short notice or even en-route [AESW12].

Technological advances, both hardware and software, are key enablers for dynamic ridesharing. The first influential fact is that the smartphones are becoming hugely popular [Ema14, Smi15]. The first impact of smartphones on ridesharing is that they provide an infrastructure on which a ridesharing application can run, replacing the old-fashioned, sometimes not so convenient, approaches such as phone or website. More importantly, smartphones are usually equipped with helpful communication capabilities, including Global Positioning System (GPS) [Zic12] and network connectivity [DS13].

Dynamic ridesharing by its nature is able to ease some aspects of existing challenges in traditional ridesharing. For example, tracking participants by means of GPS could mitigate safety concerns or increase the reliability. In terms of flexibility, since dynamic ridesharing does not necessarily require a long-term commitment, users have the option to request a trip sharing day-by-day or whenever they are pretty sure about their itinerary.

Even though the above advancement in technology could be beneficially available, ridesharing is still in short supply. Here, we focus on the ride-matching problem which is central to the concept. However, we are mindful of the fact that there are a number of other challenges that should be dealt with to accomplish the ultimate success of ridesharing.

4.3 Automated Ride-Matching

We start this section with an example that explains a simple scenario of matching between two individuals in a ridesharing application.

Example 8 » A Simple Ride-Matching Scenario

Suppose Alice is going to drive from A to B, and her driving speed is around $80^{km}/_{h}$ *on average. Another individual, Bob, needs to travel from C to D. To do this, he goes up from C to E with a taxi to get a train; then, he gets off the train at F, and finally he takes a bus to reach his destination D. His*

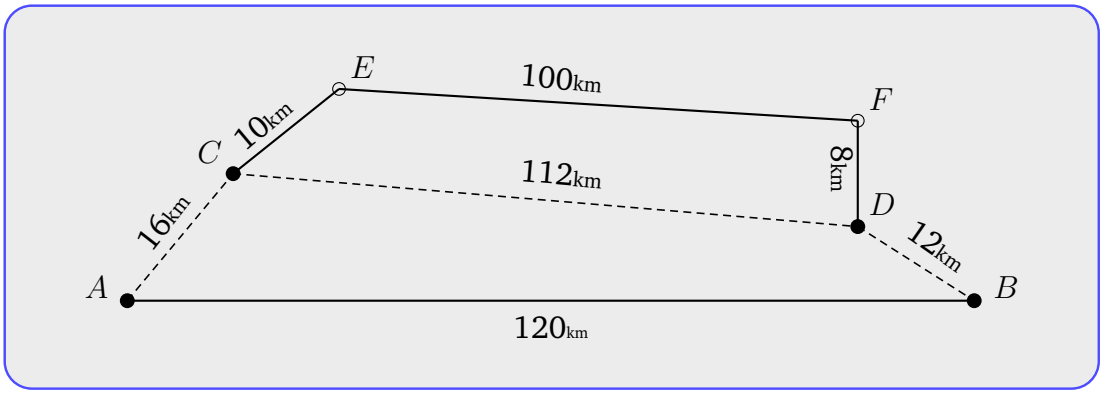

Figure 4.1: The routes of Bob and Alice's trips, described in Example 8.

trip will take approximately 115 minutes. The schema of routes and their distances are depicted in Figure 4.1.

If a ride-matching between them is suggested, Alice needs to drive to C to pick up Bob, and then drops him off at D, and proceeds to her destination B. By doing this, the total system-wide vehicle-mileage will be ($16 + 112 + 12 =$*) 140km, and the total system-wide travel time will be 189 minutes because Alice's trip will take (*$140 \times {}^{60}/_{80} =$*) 105 minutes and Bob's travel will last (*$112 \times {}^{60}/_{80} =$*) 84 minutes.*

On the other hand, if they travel individually, the total system-wide vehicle-mileage (including bus, taxi and train) will be 238km since Bob travels ($10 + 100 + 8 =$*) 118km and Alice 120km, and the total system-wide travel time will be (*$120 \times {}^{60}/_{80} + 115 =$*) 205 minutes.*

Thus, if this matching takes place, ($238 - 140 =$*) 98km travel distance will be saved, and (*$205 - 189 =$*) 16 minutes travel time will be saved. In this scenario, Alice will drive 20km (15 minutes) more than her original route (AB), which is usually compensated with a fair payment by Bob.*

The automation of the ride-matching process (e.g., between Alice and Bob) is the essential element of dynamic ridesharing. This allows trips to be arranged at short notice with minimal effort from participants; i.e., a system helps riders and drivers to find suitable matches and facilitates the communication between participants [HKLL+06, AESW12].

In order to model the matching problem, two disjoint types of ridesharing requests are considered: a set of requests in which the owner of the requests are

Table 4.1: Each cell indicates the amount of saved travel distance (i.e., d_{ij}) for a match between corresponding driver and rider, where infeasible matchings are shown with hyphen ($-$).

	$\mathbf{R_1}$	$\mathbf{R_2}$	$\mathbf{R_3}$	$\mathbf{R_4}$
$\mathbf{D_1}$	22	(21)	–	28
$\mathbf{D_2}$	14	–	10	–
$\mathbf{D_3}$	12	10	(19)	8
$\mathbf{D_4}$	(30)	–	18	–
$\mathbf{D_5}$	–	7	25	(27)

drivers (D), and requests created by riders (R). Hence, all trip requests could be represented by the set $\mathcal{S} = D \cup R$. Then, ridesharing requests are represented as a bipartite graph $G = (D, R, E)$, with E denoting the edges of the graph. This setting can be extended for the case when some participants are happy with being either a driver or a rider.

This graph becomes a weighted graph by assigning a weight c_{ij} to the edge (D_i, R_j), where $D_i, R_j \in \mathcal{S}$. Generally speaking, c_{ij} quantifies how much is gained by matching D_i and R_j. This weight is usually a composition of a system's overall benefits. For representing the system's benefits—which ultimately could result in less pollution, traffic congestion etc.,—two measures are often mentioned in the related studies; the saved travel distance (d_{ij}) and the saved travel time (t_{ij}) which are obtained from the match (D_i, R_j) [CdLHM04, WN06]. For instance, c_{ij} could be defined to be $d_{ij} + t_{ij}$ (or some other linear combination). To represent infeasibility of matching between D_i and R_j, c_{ij} could be assigned to be a very small number (e.g., $-\infty$).

For finding optimal matchings, one approach popular in the literature is solving an optimisation problem in which the sum of the benefits from the proposed matchings is maximised [AESW11]. To do this, a binary decision variable x_{ij} is introduced that would be 1 when the match (D_i, R_j) is proposed, and 0 if not (it is assumed that each driver is matched to at most one rider, and each rider is matched to at most one driver). Then, the objective function to maximise is $\sum_{i,j} x_{ij} c_{ij}$. After running the solver, a fixed schedule is proposed to users as the optimal solution.

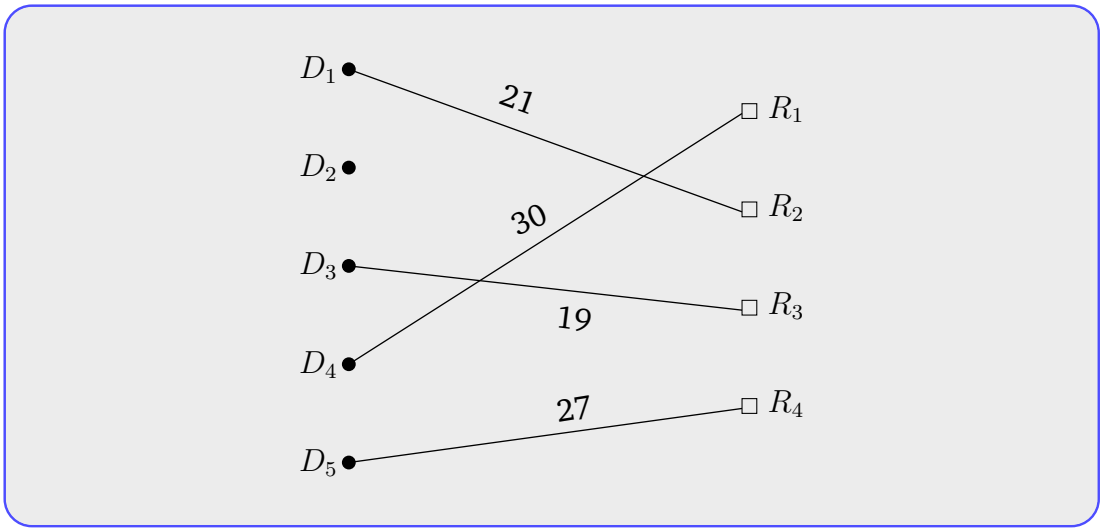

Figure 4.2: Having assumed that $c_{ij} = d_{ij}$ for each i and j in Example 9, the optimal matching is drawn. The total saved travel distance for this whole matching is 97km, which is the maximum.

Example 9 » Finding Optimal Ride-Matching

Consider a situation where there are potentially five drivers and four riders to share their journeys. So, $i \in \{1, \ldots, 5\}$ and $j \in \{1, \ldots, 4\}$. Table 4.1 shows how many kilometres would be saved by sharing rides between each pair of individuals, where infeasible matchings are represented by a hyphen (e.g., $d_{2,1} = 14$). If we consider $c_{ij} = d_{ij}$ for each i and j in this case, solving a maximisation problem as described above leads to the matching graph drawn in figure 4.2. In this example, the maximum of total saved distance is 97km.

The maximisation approach, however, neglects a crucial requirement of a practical system, that is, getting users' confirmation before fixing a ride-share for them. Although earlier we emphasised the concept of automation in ride-matching which helps to minimise users' efforts, we believe that it could not be fully automatic. In fact, it is hard in practice to convince users to share their ride with somebody without their final agreement. For example, regarding Figure 4.2, it does not make sense to send a message to D_1 saying: "According to our computation, the best match for you is R_2. So, you are supposed to give him a ride tomorrow morning at 8:30."

For this reason, we suggest a novel attitude towards ride-matching problems, by looking at the problem as a recommendation system rather than an optimisation problem. In this setting, the system just recommends a set of best possible matchings to each individual with respect to the weights c_{ij}.

As previously mentioned, c_{ij} only considers the system's benefits, and to express users' limitations, the common strategy is posing some constraints on the matching graph [HW12, AESW11]. Nevertheless, there are two main shortcomings that have not been fully addressed by adoption of this strategy:

Only Hard Constraints The fact is that users might sacrifice some of their desires in favour of other ones. For instance, a rider who likes smoking in the car may decide to share his trip in a non-smoking vehicle due to other favourable conditions. Therefore, having soft constraints instead of hard ones may often be more plausible. Posing soft constraints could be rephrased as considering users' preferences.

Eliciting Complicated Parameters In order to set constraints, there is a prerequisite to ask users to specify several parameters explicitly, such as earliest possible departure time, latest possible arrival time and maximum excess travel time [BMM04, AESW11, AAM11]. However, elicitation of such parameters for every trip is quite impractical in real-world situations. As an example, users can simply state their departure time, but finding how flexible they are with that time is not a straightforward task. Moreover, some participants may be hesitant or unwilling to disclose certain preferences for privacy reasons.

Our solution for attempting to overcome the aforementioned issues is redefining c_{ij} in such a way that not only does c_{ij} incorporate the overall system's benefits, but also takes into account participants' preferences. Actually, c_{ij} should also represent how happy two individuals (D_i and R_j) are with being matched with each other. While positional and temporal elements of a ride-share usually play a key role in forming users' satisfaction degree, users' social preferences such as the other party's reputation and gender, smoking habit and so forth are also suggested as relevant components [GHH11]. To illustrate mathematically, assume $c_{ij} = (w_{ij}^{(D)} \cdot w_{ij}^{(R)}) \cdot (d_{ij} + t_{ij})$ where $w_{ij}^{(D)}$ quantifies how much the driver i would like to share the journey with the rider j, and $w_{ij}^{(R)}$ indicates how favourable sharing the ride with the driver i is for the rider j; if either $w_{ij}^{(D)}$ or $w_{ij}^{(R)}$ is negative then $c_{ij} = -\infty$. In order to find $w_{ij}^{(D)}$ (resp. $w_{ij}^{(R)}$), we suggest learning the preferences of the driver i (resp. the rider j) from his/her past

ridesharing records. For instance, we can learn from the previous choices of a particular user that he is normally more flexible with time than location. We will discuss more about this subject in Section 4.3.1. As a result of learning weights, as well as modelling soft constraints in the form of users' preferences, there is no need to elicit those cumbersome parameters from participants anymore.

4.3.1 Learning Weights

In this section, the mechanism of learning user preferences from the previous ridesharing records will be characterised. To give an intuitive sense, we start with an example.

Example 10 » Learning Weights Mechanism

Take Alice from Example 8 who has a daily driving trip from A to B (e.g., every day at 8:30). This trip request is denoted by D_1.

On Day 1, the system recommends three matching opportunities, namely R_1, R_2 and R_3. Alice declines R_1, accepts R_2 and leaves R_3 without any response. Ignoring a recommended match may potentially mean that the user had been tentative about it.

On the second day, two recommendations, R_4 and R_5, are suggested to her. R_4 is declined and R_5 is initially accepted. While Alice was waiting for the response of the other party in R_5, she receives another suggestion, R_6; she then changes her mind about R_5 and cancels that one, and instead accepts R_6.

On the third day, the system has found two feasible matches which are R_7 and R_8. The goal is to evaluate how desirable these two opportunities are for Alice regarding her trip D_1. Using the notation of the previous section, we would like to find $w_{1,7}^{(D)}$ and $w_{1,8}^{(D)}$ which are finally incorporated in the calculation of $c_{1,7}$ and $c_{1,8}$, respectively. Table 4.2 summarises this example.

At first glance, the problem might seem to be a classification problem because the supervision (labels) is in the form of classes (i.e., Accepted, Declined etc.). However, a closer look suggests that learning a classifier may well not be a good

Table 4.2: The scenario described in Example 10 is summarised here. Alice's responses are Accepted (A), Cancelled (C), Ignored (I) and Declined (D).

Day 1	Day 2	Day 3 (Today)
(R_1, D)	(R_4, D)	$R_7(w_{1,7}^{(D)} = ?)$
(R_2, A)	(R_5, C)	$R_8(w_{1,8}^{(D)} = ?)$
(R_3, I)	(R_6, A)	

choice. The first reason is that there is a logical order between classes in the current case (e.g., Accepted has the highest value and Declined the lowest value), whereas classification is often applied when there is no ordering between class labels. Secondly, a classifier will predict a class label whereas here, a scalar value is required.

If each class label is replaced by an appropriate number which keeps the natural ordering, the problem could be considered as a regression problem. For instance, the number 1 might be assigned for Accepted, 0.67 for Cancelled, 0.33 for Ignored and 0 for Declined. In spite of the fact that learning a regressor has none of those defects mentioned about classification, we believe that it still suffers from the following flaws:

- The user's response to a recommended opportunity not only depends on the properties of that particular case, but also depends on other rival opportunities. Taking Example 10 to illustrate, it is not necessarily the case that if R_2 existed in the suggestion list on day 2, it would certainly be accepted again, because Alice accepted R_2 in presence of R_1 and R_3 which does not guarantee its acceptance when R_5 and R_6 are available.
- Two class labels do not necessarily have the same desirability distance for all instances. Consider Example 10 again; replacing class labels with numbers as described above suggests that the difference between R_1 and R_2 from the first day, and R_4 and R_6 from the second day are both 1. However, the only thing that is known is that Alice preferred R_2 to R_1 and R_6 to R_4, not the extent of the difference.

To address the above issues, we suggest considering the supervision in the form of qualitative preferences. This means that a set of *pairwise comparisons* among

alternatives is derived from the user's choices. The following set of preferences could be formed from information given in Example 10:

$$\{(R_2 \succ R_3), (R_3 \succ R_1), (R_2 \succ R_1), (R_6 \succ R_5), (R_5 \succ R_4), (R_6 \succ R_4)\}.$$

In the next section, a model will be developed to learn a utility function by making use of this kind of preferences set.

4.4 Learning Model: SVPL

In this section, we develop a method which is referred to as *SVPL* (Support Vector Preferences Learner) in our experiments.

4.4.1 Basic Formulation

As described above, the primary capability of the method being developed in this section should be learning a scoring function from a set of pairwise comparisons, expressed between several alternatives. This function can generate a scalar number for each alternative (i.e., ridesharing trip opportunity), measuring the expected desirability degree of that alternative for the user.

We assume that some user has told us (explicitly or implicitly) that she prefers feature vector $a_i \in \mathbb{R}^n$ over $b_i \in \mathbb{R}^n$, for each $i \in I = \{1, \ldots, m\}$. Each tuple a_i or b_i in $\mathbb{R}^n$ represents an alternative that is characterised by n features, with $a_i(k)$ being the score for alternative a_i regarding the kth feature (each feature representing a different property of the trip). Features are assumed to be numeric, but again ordinal and nominal features can also be considered with the same approaches explained in Section 2.3.

Now, the goal is finding a function $f : \mathbb{R}^n \to \mathbb{R}$ which maps a features vector to a scalar value and is in agreement with the preferences set, i.e., we aim to have $f(a_i) > f(b_i)$ for all $i \in I$. In order to achieve this goal, we developed a derivation of conventional SVM which eventually turns out to be similar to the SVMRank approach proposed in [Joa02]. In the current section, we assume that there exists such a function meaning that preferences relations are consistent. Another assumption being made in this section is that the function is linear.

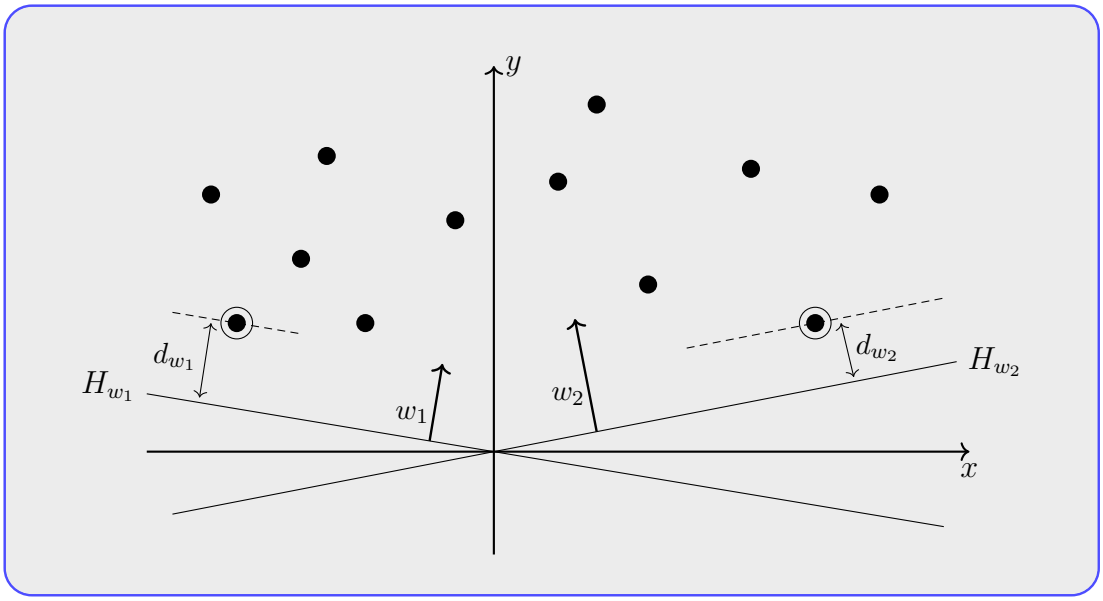

Figure 4.3: Two samples of plausible hyperplanes (H_{w_1} and H_{w_2}) and their associated normal vectors (w_1 and w_2) for a set of consistent preferences inputs are illustrated. d_{w_1} and d_{w_2} are the margins of H_{w_1} and H_{w_2}, respectively.

we consider the inconsistent case and the non-linear case in Sections 4.4.2 and 4.4.3 respectively. Hence, defining $f(x) = x \cdot w$ leads to:

$$\forall i \in I, \quad a_i \cdot w > b_i \cdot w \tag{4.1}$$

where $w \in \mathbb{R}^n$ is an unknown weighting vector.

We define Λ, the *preference inputs*, to be $\{\lambda_i : i \in I\}$, where for each i, $\lambda_i = a_i - b_i$. Thus, with respect to those two assumptions, there exists $w \in \mathbb{R}^n$, such that $\lambda \cdot w > 0$ for all $\lambda \in \Lambda$ (because $a_i \cdot w > b_i \cdot w$). We can associate the hyperplane $H_w = \{x \in \mathbb{R}^n : x \cdot w = 0\}$ with a feasible w (w is the normal vector to the hyperplane H_w). Clearly, any feasible hyperplane contains the origin, and all $\lambda \in \Lambda$ are in the associated positive open half-space of the hyperplane. Two feasible hyperplanes (H_{w_1} and H_{w_2}) for a consistent set of preferences inputs are depicted in Figure 4.3.

Similar to conventional SVM, the *margin* of a hyperplane, denoted by d_w, is defined as the distance from the hyperplane to the closest preference input. The distance from λ to H is $\frac{w \cdot \lambda}{\|w\|}$; so, the margin is equal to:

$$d_w = \min_{\lambda \in \Lambda} \frac{w \cdot \lambda}{\|w\|}. \tag{4.2}$$

The following lemma provides a simpler formulation for margin.

Lemma 1: Margin Simplification

Consider any $w \in \mathbb{R}$ such that $w \cdot \lambda > 0$ for all $\lambda \in \Lambda$. Define a_w to be $\min_{\lambda \in \Lambda} w \cdot \lambda$, and define $\bar{w}$ to be $\frac{w}{a_w}$. Then, the following all hold.

(i) $\bar{w} \cdot \lambda \geq 1$ for all $\lambda \in \Lambda$;

(ii) if $w \cdot \lambda \geq 1$ for all $\lambda \in \Lambda$, then $\|w\| > \|\bar{w}\|$ unless $w = \bar{w}$;

(iii) for any real $r > 0$, $d_{rw} = d_w$;

(iv) $d_w = d_{\bar{w}} = \frac{1}{\|\bar{w}\|}$;

(v) $H_w = H_{\bar{w}}$.

Proof: $a_{\bar{w}} = \min_{\lambda \in \Lambda} \frac{1}{a_w} w \cdot \lambda = \frac{a_w}{a_w} = 1$. Thus, $\bar{w} \cdot \lambda \geq 1$ for all $\lambda \in \Lambda$, showing (i). Also, $\frac{\|w\|}{\|\bar{w}\|} = a_w$, by definition of $\bar{w}$. If $w \cdot \lambda \geq 1$ for all $\lambda \in \Lambda$ then $a_w \geq 1$, so $\|w\| > \|\bar{w}\|$ unless $a_w = 1$, i.e., $w = \bar{w}$, proving (ii). The definitions immediately imply that $d_w = \frac{a_w}{\|w\|}$. Since $a_{\bar{w}} = 1$, we have $d_{\bar{w}} = \frac{1}{\|\bar{w}\|}$. The definition of d_w implies that for any real $r > 0$, $d_{rw} = d_w$, showing (iii), so, in particular, $d_w = d_{\bar{w}} = \frac{1}{\|\bar{w}\|}$, which proves (iv). Since a_w is strictly positive, the definition of H_w implies (v). ■

Like SVM, it seems reasonable that a greater marginal distance from the condition boundary is more desirable. Thus, among all feasible hyperplanes, we look for the hyperplane that produces the largest margin. Consider $w \in \mathbb{R}^n$ that represents this hyperplane with the maximal margin; then Lemma 1(v) implies that $H_{\bar{w}} = H_w$. Thus, we need to maximise the margin stated in Lemma 1(iv) subject to the constraints stated in Lemma 1(i). That leads, by replacing $\bar{w}$ with

w, to the following optimisation problem:

$$\underset{w}{\text{maximise}} \quad \frac{1}{\|w\|} \tag{4.3a}$$

subject to

$$w \cdot \lambda \geq 1 \qquad \forall \lambda \in \Lambda. \tag{4.3b}$$

To have a more standard form, the arrangement of the problem is reformulated as a minimisation problem in this manner:

$$\underset{w}{\text{minimise}} \quad \frac{1}{2}\|w\|^2 \tag{4.4a}$$

subject to

$$w \cdot \lambda_i \geq 1 \qquad \forall i \in I. \tag{4.4b}$$

Similar to conventional SVM optimisation (see Equation 2.7 in Chapter 2), this result is a convex quadratic programming problem. Thus, a QP solver can be exploited to find the weighting vector w.

Now, we switch to the *Lagrangian Dual Problem* in which preferences inputs only appear in the form of pairwise dot products (both in the objective function and constraints). In Section 4.4.3, more explanation will be given as to why we are interested in this form of problem formulation. For our problem, the *Lagrangian function* which is basically obtained by augmenting the objective function with a weighted sum of the constraint functions, is:

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + \sum_{i \in I} \mu_i \, (1 - w \cdot \lambda_i), \tag{4.5}$$

where μ_i is referred to as the *Lagrange multiplier* associated with the i^{th} inequality constraint $w \cdot \lambda_i \geq 1$ [BV04, Chapter 5].

The optimisation problem stated in Equation (4.4) is called the primal form;

where the Lagrangian dual form is formulated in this fashion:

$$\underset{\mu_i}{\text{maximise}} \quad \inf_{w} \mathcal{L} \tag{4.6a}$$

subject to

$$\mu_i \geq 0 \qquad \forall i \in I. \tag{4.6b}$$

Because the primal form is convex and the Slater[1] condition holds, we say the *strong duality* holds for our problem [BV04, Sec. 5.2.3]. This means that the optimal value for the dual problem equals the optimal value for the primal form. As a result, solving the Lagrangian dual form (Equation 4.6) is equivalent to solving the primal form of the problem (Equation 4.4).

As stated, strong duality is obtained for our problem, and also the objective function is differentiable. Regarding these two criteria, an optimal value must satisfy the *Karush Kuhn Tucker (KKT)* conditions [BV04, Sec. 5.5.3]. Here, we just make use of the *Stationarity* condition of KKT which states[2]:

$$\begin{aligned} \text{for each component } j \in \{1,\dots,n\}, \quad & \frac{\partial \mathcal{L}}{\partial w(j)} = 0 \\ & w(j) - \sum_{i \in I} \mu_i \lambda_i(j) = 0 \\ \Rightarrow \quad & w = \sum_{i \in I} \mu_i \lambda_i. \end{aligned} \tag{4.7}$$

[1] Consider there are m constraints in the form of $f_i(x) \leq 0$ where $i \in \{1,\dots,m\}$. The Slater condition holds if and only if there exists $x^* \in \bigcap_i dom(f_i)$ such that $f_i(x^*) < 0$ for all $i \in \{1,\dots,m\}$. For our case, $f_i(\mu_i) = -\mu_i$, so, choosing μ^* such that $\mu^* > 0$ (e.g., $\mu^* = 1$) satisfies the condition.

[2] In [BV04, Sec. 5.5.3], the stationarity condition corresponds to $\triangledown f_0(\tilde{x}) + \sum_{i=1}^{m} \tilde{\lambda_i} \triangledown f_i(\tilde{x}) + \sum_{i=1}^{p} \tilde{v_i} \triangledown h_i(\tilde{x}) = 0$, where $f_0(x)$ is the objective function, $f_i(x)$ for $i \in \{1,\dots,m\}$ are inequality constraints, $h_i(x)$ for $i \in \{1,\dots,p\}$ are equality constraints, and $\tilde{\lambda_i}, \tilde{v_i} \in I\!R$ are KKT multipliers. In our case, since we don not have equality constraints and there is no w in the inequality conditions (i.e., Equation 4.6b), only the first part of the left side of the equation remains which is the objective function (i.e., $\mathcal{L}$). This immediately leads to Equation 4.7.

Making use of Equation 4.7, we can rewrite Equation 4.5 without w:

$$\begin{aligned}\mathcal{L} &= \frac{1}{2} w \cdot w + \sum_{i \in I} \mu_i - \sum_{i \in I} \mu_i w \cdot \lambda_i \\ &= \frac{1}{2} (\sum_{i \in I} \mu_i \lambda_i) \cdot (\sum_{j \in I} \mu_j \lambda_j) + \sum_{i \in I} \mu_i - \sum_{i \in I} \mu_i (\sum_{j \in I} \mu_j \lambda_j) \cdot \lambda_i \\ &= \frac{1}{2} \sum_{i \in I} \sum_{j \in I} \mu_i \mu_j \lambda_i \cdot \lambda_j + \sum_{i \in I} \mu_i - \sum_{i \in I} \sum_{j \in I} \mu_i \mu_j \lambda_i \cdot \lambda_j \\ &= \sum_{i \in I} \mu_i - \frac{1}{2} \sum_{i \in I} \sum_{j \in I} \mu_i \mu_j \lambda_i \cdot \lambda_j.\end{aligned}$$

This leads to the following version of the problem in which the preferences inputs only appear in the form of dot products (i.e., $\lambda_i \cdot \lambda_j$).

$$\underset{\mu}{\text{maximise}} \qquad \sum_{i \in I} \mu_i - \frac{1}{2} \sum_{i \in I} \sum_{j \in I} \mu_i \mu_j \lambda_i \cdot \lambda_j \tag{4.8a}$$

subject to

$$\mu_i \geq 0 \qquad \forall i \in I. \tag{4.8b}$$

Solving this new form of the problem gives optimal values for the Lagrange multipliers (μ); we can then utilize Equation 4.7 to find w as well.

4.4.2 Handling Inconsistencies

So far, we have assumed that there exists at least one hyperplane such that all preferences inputs are placed in the positive open half-space of it. However, it is possible that in practice this assumption may result in finding no feasible solution. To handle inconsistencies, we reformulate the initial constraint (4.4b) such that it could be violated, but with a cost.

For this purpose, the constraint (4.4b) is rewritten as $1 - w \cdot \lambda_i \leq \xi_i$ where ξ_i is a slack variable measuring the extent to which the i^{th} constraint is violated, and obviously should be non-negative ($\xi_i \geq 0$). Then, the objective function is augmented by the term $C \sum_{i \in I} \xi_i$ to reveal the effect of costs, where C is a constant parameter to be adjusted by the user. The parameter C scales the impact of inconsistent points; a larger C corresponds to assigning a higher penalty to

errors. The primal form of the problem becomes the following:

$$\underset{w,\xi}{\text{minimise}} \qquad \frac{1}{2}\|w\|^2 + C\sum_{i\in I}\xi_i \tag{4.9a}$$

subject to

$$1 - w\cdot\lambda_i \leq \xi_i \qquad \forall i \in I, \tag{4.9b}$$

$$\xi_i \geq 0 \qquad \forall i \in I. \tag{4.9c}$$

Because we have a new set of constraints ($\xi_i \geq 0$), a new set of positive Lagrange multipliers α_i are introduced. The Lagrangian changes as follows:

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + C\sum_{i\in I}\xi_i + \sum_{i\in I}\mu_i\left(1 - w\cdot\lambda_i - \xi_i\right) - \sum_{i\in I}\alpha_i\xi_i \tag{4.10}$$

The stationarity KKT condition entails the additional equality constraints:

$$\begin{aligned}\text{for all } i \in I, \quad \frac{\partial\mathcal{L}}{\partial\xi_i} &= 0,\\ &\Rightarrow C - \mu_i - \alpha_i = 0,\\ &\Rightarrow \alpha_i = C - \mu_i.\end{aligned}$$

By substituting $C - \mu_i$ for α_i in Equations 4.10, we have:

$$\begin{aligned}\mathcal{L} &= \frac{1}{2}\|w\|^2 + C\sum_{i\in I}\xi_i + \sum_{i\in I}\mu_i\left(1 - w\cdot\lambda_i - \xi_i\right) - \sum_{i\in I}(C-\mu_i)\xi_i\\ &= \frac{1}{2}\|w\|^2 + C\sum_{i\in I}\xi_i + \sum_{i\in I}\mu_i\left(1 - w\cdot\lambda_i\right) - \sum_{i\in I}\mu_i\xi_i - C\sum_{i\in I}\xi_i + \sum_{i\in I}\mu_i\xi_i\\ &= \frac{1}{2}\|w\|^2 + \sum_{i\in I}\mu_i\left(1 - w\cdot\lambda_i\right).\end{aligned}$$

This result is the same as the Lagrange form in Equation 4.5. So, as in the previous section, replacing w with $\sum_{i\in I}\mu_i\lambda_i$ gives us Equation 4.8a for $\mathcal{L}$. Also, since $\alpha_i \geq 0$ for all $i \in I$, we have $C - \mu_i \geq 0$ and hence $\mu_i \leq C$. That leads to the Lagrangian dual form of the problem as follows:

$$\underset{\mu}{\text{maximise}} \qquad \sum_{i\in I}\mu_i - \frac{1}{2}\sum_{i\in I}\sum_{j\in I}\mu_i\mu_j\lambda_i\cdot\lambda_j \tag{4.11a}$$

subject to

$$0 \leq \mu_i \leq C \qquad \forall i \in I. \tag{4.11b}$$

Since neither the ξ_i nor its associated Lagrange multiplier (α_i) appear in the objective function, this format of the problem is very similar to the one without introducing costs (Equation 4.8), except that μ_i is restricted by the upper bound C.

4.4.3 Non-Linear Utility Functions

We assumed that the utility function is a linear function. However, a non-linear scoring function might be a better choice sometimes. In this section, we deal with this matter to cover the non-linear case as well.

The *Kernel function* concept [ABR64] is a widely-used trick for pattern recognition problems which can be also used for our case. The idea comes from the fact that a set of non linearly-representable data could be managed in a linear fashion if mapped to a higher dimension. To do the mapping, we assume a function Φ of the form:

$$\Phi : I\!R^n \rightarrow \mathcal{H}, \tag{4.12}$$

where $\mathcal{H}$ denotes a higher dimensional space than n-dimensional. If all preferences inputs are mapped into $\mathcal{H}$ by making use of Φ, the Lagrangian function will be:

$$\mathcal{L} = \sum_{i \in I} \mu_i - \frac{1}{2} \sum_{i \in I} \sum_{j \in I} \mu_i \mu_j \Phi(\lambda_i) \cdot \Phi(\lambda_j). \tag{4.13}$$

A deeper look into the process of computing $\mathcal{L}$ (i.e., Equation 4.13) reveals that, even though $\mathcal{L}$ is still a scalar value, the optimisation problem performs a dot product operation ($\Phi(\lambda_i) \cdot \Phi(\lambda_j)$) in the high dimensional space, which is computationally expensive, if the dimension of $\mathcal{H}$ is extremely large.

Principally, a kernel is a function which operates in the lower dimension, i.e., $K : I\!R^n \times I\!R^n \rightarrow I\!R$, but yields an identical result to the dot product of mapped vectors in the higher dimension, i.e., $K(X, Y) = \Phi(X) \cdot \Phi(Y)$.

Due to the above property of the kernel function, the term $\Phi(\lambda_i) \cdot \Phi(\lambda_j)$ can be simply replaced in Equation (4.13) with an appropriate kernel. The great advantage of such a replacement is that the complexity of the optimisation problem remains only dependent on the dimensionality of the lower dimensional space (i.e., n) and not of $\mathcal{H}$.

Note that this simplification happens without even explicitly stating the Φ function because the problem has been formulated in terms of dot products of

points. So, the problem is rewritten as follows:

$$\underset{\mu}{\text{maximise}} \qquad \sum_{i\in I}\mu_i - \frac{1}{2}\sum_{i\in I}\sum_{j\in I}\mu_i\mu_j K(\lambda_i,\lambda_j) \tag{4.14a}$$

subject to

$$0 \leq \mu_i \leq C \qquad \forall i \in I. \tag{4.14b}$$

There still remains one point that should be considered. Solving this optimisation problem only gives μ where after all, w is required. Equation 4.7 cannot be used to obtain w anymore, because after the mapping, w will live in $\mathcal{H}$; that is:

$$w = \sum_{i\in I}\mu_i\Phi(\lambda_\mathbf{i}). \tag{4.15}$$

However, recall that finding w is just an intermediate goal to achieve the utility function. Therefore, using Equation (4.15) brings the following form of the utility function:

$$\begin{aligned} f(x) &= w\cdot\Phi(x) \\ &= (\sum_{i\in I}\mu_i\Phi(\lambda_i))\cdot\Phi(x) \\ &= \sum_{i\in I}\mu_i(\Phi(\lambda_i)\cdot\Phi(x)) \\ &= \sum_{i\in I}\mu_i\ K(\lambda_i,x). \end{aligned} \tag{4.16}$$

Equations 4.14 and 4.16 form the method, which is referred to as *SVPL* (Support Vector Preferences Learner) in our experiments. As seen, the input parameters that should be chosen in SVPL are C and the kernel function's parameters (if it has any).

4.5 Experiments

4.5.1 Data Repository

The experiments make use of a subset of a year's worth of real ridesharing records, provided by Carma[3] (formerly Avego). Carma is a software company

[3] `https://www.gocarma.com/`

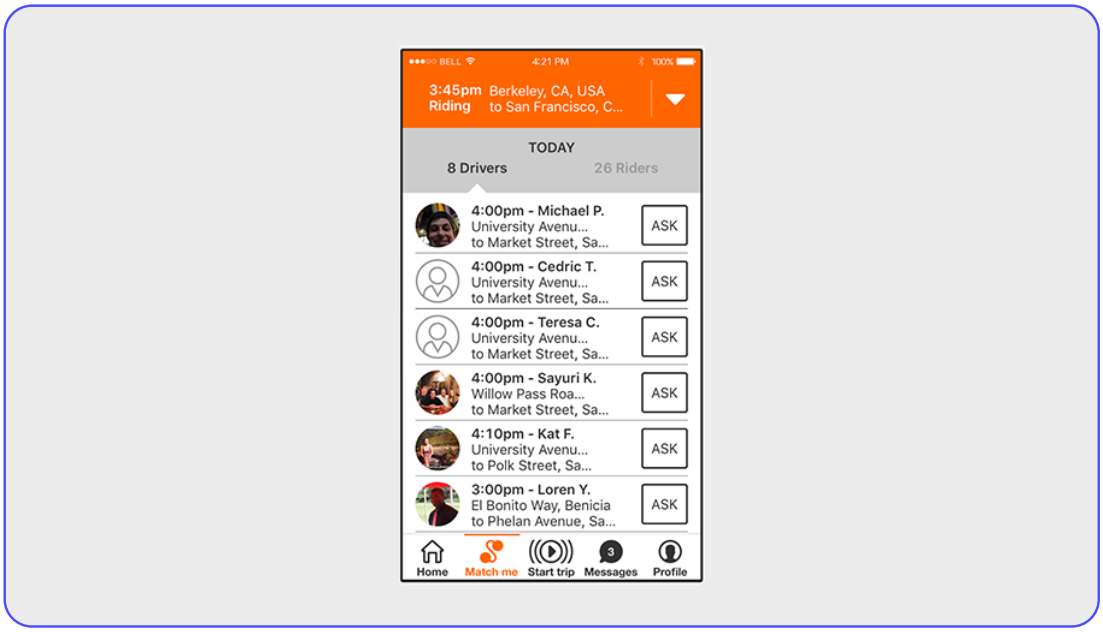

Figure 4.4: A screenshot of Carma's application to offer a ride-match.

that was offering (until 2016) a dynamic ride-share application for internet-enabled mobile phones.

The process of Carma's application is briefly explained so as to facilitate the understanding of the structure of the provided database. Users can request a ride-share as a driver, rider or both (happy to be rider or driver). Once a ridesharing request is created, the system finds and shows a suggestion list of matching opportunities on the user's screen. For each suggested item, the user can accept, decline or just leave it with no response. Note that accepting an opportunity at this stage does not mean the ride-share will certainly happen because it also needs the confirmation of the other party; so, it is observed in the database that for a single trip a user may have several accepted items. While the system is waiting to get the second confirmation, the user can cancel an initially accepted item. It should be pointed out that the cancellation of a specific item differs from the cancellation of the whole trip request; the former often happens because of the greater attractiveness of a new item, and the latter is for the time when the user does not want to share the ride for the created trip anymore.

According to the above mechanism, a *ride-matching record* is associated with a class label which is among these four: *Accepted*, *Cancelled*, *Ignored* or *Declined*. In fact, the class label indicates the response of the *main* user towards sharing the ride with the *target* user.

The second element of a ride-matching record is a vector of features, built from the personal information of both the main and the target user, and the properties of their trips. Before starting the learning process, normalizing the features' spaces is an essential requirement due to the fact that margin based methods are known to be sensitive to the way features are scaled [BHW10]. Because of that, features are scaled to an identical range, i.e., $[0, 1]$. The extracted features from the provided database are listed here:

- Positional Component: Expressing how suitable the pick-up and drop-off locations will be for the main user.
- Temporal Component: Expressing how appropriate the departure time will be for the main user.
- Gender Component: Indicating whether the target user is of the same gender (1) or the opposite (0).
- Has Image: It considers whether the target user's profile contains his/her picture or not.
- Average Rating: At the end of a ridesharing experience, users can rate the other party. This feature holds the average rate that the target user has gained from his/her previous trips.
- Is Favourite: As well as the rating, a user can mark a particular user as a favourite person at the end of ridesharing. This feature shows whether the target user is among individuals who are marked as a favourite of the main user or not.
- Previous Rating: If the main user and the target user have had a previous ridesharing experience with each other, this feature shows the rating that the main user has given to the target user.

We base our experiments on 12 benchmarks derived from this data-set. Each benchmark corresponds to a different user (who is the main user of all ride-matching records of that benchmark). Table 4.3 shows the number of ride-matching records, separated by class labels, for each benchmark.

4.5.2 Experiments Settings

For each benchmark, the ride-matching records are sorted in chronological order of the creation time (i.e., earliest first), and then split into two parts. The

Table 4.3: 12 benchmarks derived from a commercial ridesharing system which are being used in the experiments.

Benchmark	Accepted	Cancelled	Ignored	Declined	**Total**
1.	179	72	83	14	348
2.	34	1	49	0	84
3.	210	52	138	2	402
4.	3	1	124	41	169
5.	81	1	53	29	164
6.	126	3	74	45	248
7.	39	7	28	4	78
8.	45	7	78	29	159
9.	146	37	3	1	187
10.	126	24	14	1	165
11.	40	1	41	13	95
12.	96	12	25	4	137
Total	1125	218	710	183	2236

first part includes 80% of the records which we use to derive a set of pairwise comparisons as explained in 4.3.1. This set works as input data to train SVPL. At the end of the learning stage, SVPL can predict, for each record, a scalar value which expresses the utility of that record for the user.

The second part of the data is utilized for the testing stage. From this data, a total pre-order between the records with respect to class labels could be derived (see Example 11 below). This ranking is used as the ground truth for testing. On the other hand, the predicted scalar values for records produced by the model generates a total pre-order between records.

Thereafter, the *C-Index* (or concordance *C*) measure is exploited so as to assess the ranking performance of SVPL with respect to the ground truth ranking [GH05]. It is calculated as follows:

$$\text{C-Index}(r, \hat{r}) = \frac{\kappa}{\kappa + \bar{\kappa}}$$

where κ is the number of correctly ranked pairs and $\bar{\kappa}$ (Kendall tau distance) is the number of incorrectly ranked pairs.

DCG is another metric that is used to evaluate the model's performance [JK02]. To compute DCG, first each item is assigned with a relevance degree; items that are supposed to be in higher ranks have greater relevance degree. Here, we say that the relevance degree of an accepted ride-matching is 3; a cancelled one is

2; ignored is 1 and declined is 0. Then, for a particular ranking among p items, DCG is defined as follows[4]:

$$\text{DCG}_p = \frac{rel_1}{0.4} + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$$

where rel_i is the relevance degree of the item at rank i. Since DCG_p also depends on p, it should be normalized. Thus, we use normalized DCG (nDCG) which is calculated by dividing "DCG of the current ranking" by "DCG of the ideal ranking".

Example 11 » Ranking Metrics

Suppose there are 6 ride-matching opportunities for Alice's trip D_1 as stated in Table 4.4. From her responses (i.e., accepted, cancelled, ignored and declined) the following total pre-order can be derived as ground truth.

$$R_1, R_2 \succ R_3 \succ R_4, R_5 \succ R_6$$

Note that R_1 and R_2 are incomparable, as are R_4 and R_5. This is the ideal ranking of available options for this trip and its DCG is:

$$\frac{3}{0.4} + \frac{3}{\log_2 2} + \frac{2}{\log_2 3} + \frac{1}{\log_2 4} + \frac{1}{\log_2 5} + \frac{0}{\log_2 6} = 12.69.$$

Assume that SVPL has been previously trained by some ridesharing records of Alice, and now predicts those utilities in the 4th column. As a result, SVPL ranks options in this fashion:

$$R_2 \succ R_3 \succ R_1 \succ R_5 \succ R_6 \succ R_4$$

Given this ranking, DCG is:

$$\frac{3}{0.4} + \frac{2}{\log_2 2} + \frac{3}{\log_2 3} + \frac{1}{\log_2 4} + \frac{0}{\log_2 5} + \frac{1}{\log_2 6} = 12.28,$$

and subsequently nDCG$= {}^{12.28}/_{12.69} = 0.9677$. With respect to the C-Index metric, the ranking accuracy is 84.61%, since only two pairs–(R_1, R_3) and

[4]The original formula in [JK02]is $\text{DCG}_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$. It can be seen that the coefficient of the first term (rel_1) and the second term's (rel_2) are equal. To give a smaller penalty value to the first term, we divided rel_1 by 0.4 (instead of 1).

Table 4.4: Six ride-matching opportunities available for Alice's trip D_1, which are being used in Examples 11 and 12.

	Ground Truth		SVPL Prediction		Saved Travel Distance	
	Labels	Ranking	Utility	Ranking	Value(km)	Ranking
R_1	A	1	8.2	3	32	6
R_2	A	1	**10**	1	36	3
R_3	C	3	8.5	2	35	4
R_4	I	4	6.9	6	38	2
R_5	I	4	7.5	4	**40**	1
R_6	D	6	7	5	34	5

(R_4, R_6)–out of 13 pairs were ranked incorrectly and $^{11}/_{13} = 0.8461$.

In terms of choosing the kernel function, although there are many kernels available in the literature, and devising a new kernel by meeting *Mercer's Condition* is possible [SHS01], we just simply use three well-known kernels, listed here:

- $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$: The linear kernel which is equivalent to the non-kernel based formulation (Equation 4.11).
- $K(\mathbf{x}, \mathbf{y}) = (\gamma \mathbf{x} \cdot \mathbf{y} + r)^p$: The polynomial kernel of degree p; here, we assume $p = 2$, $\gamma = 1$ and $r = 0$.
- $K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$: This is the Gaussian radial basis function (RBF). The parameter σ is tuned as an input parameter.

For adjusting the input parameters, i.e., the errors' multiplier C and the kernel's parameters, we use the *Grid Search* algorithm for hyper-parameter optimisation [BB12]. Roughly speaking, in the grid search, the combination of input parameters is chosen in which the model performs best.

Each run of SVPL, which comprises learning and testing phases for a benchmark, takes less than a couple of minutes, making use of a computer facilitated by a Core i7 2.60 GHz processor and 8 GB RAM memory.

4.5.3 SVPL Versus Maximising Saved Travel Distance

We stated in Section 4.3 that c_{ij} is a numeric representation to show how good the matching is between individuals i and j, which mainly considers only the

benefits to the whole system such as the total saved travel distance. However, we argued that this is insufficient and c_{ij} should also take into account user preferences. Then, we suggested making use of this measure to rank and recommend best ridesharing opportunities to the user. We claim this strategy would boost the users' satisfaction degree, which should increase the chances of repeat usage of the system.

Now, we examine the importance of the user preferences by comparing the effectiveness of the ranking provided by SVPL, which involves learning user preferences, with the case in which options are ranked regardless of the user's desires; i.e., an option A precedes another option B when the benefits obtained by that option for the whole system are more. In our experiments, we especially consider the saved travel distance measure as the benefit gained for the whole system. The following example clarifies the situation.

Example 12 » Experiments Approach

Consider the situation explained in Example 11. Now, let us sort objects according to the travel distance that would be saved by each matching (see the 6th column in Table 4.4). This leads to this (total) order of objects:

$$R_5 \succ R_4 \succ R_2 \succ R_3 \succ R_6 \succ R_1$$

It should be noted again that for proposing this ranking, we set aside Alice's preferences which might be implicitly stated in her past ridesharing records; therefore, unlike SVPL, there is no need to have a learning phase. For this instance of ranking, nDCG and C-Index are respectively 0.5959 *and* 0.3846*, which both are smaller than SVPL's. From this point onwards, we call this strategy of ranking ridesharing opportunities (regarding saved travel distance) as* STD *for the sake of brevity.*

In this example, if the top recommended option suggested by SVPL (i.e., R_2*) is chosen by Alice to form a ride-share for* D_1*, then the saved travel distance will be* 36*km which is* 4*km less than the optimal matching (i.e.,* R_5*). Thus, roughly speaking, we can say by use of SVPL strategy in ranking rather than STD , the saved travel distance would probably decline by* 10%*, compared to the optimal case, in this example.* □

Table 4.5: These experimental results aim to compare SVPL against STD, based on ranking accuracy (first four columns), and the saved travel distance.

Bench.	nDCG		C-Index		Saved Travel Distance (Km)		
	STD	SVPL	STD	SVPL	STD	SVPL	Loss Rate (%)
1.	80.51	**94.81**	39.10	**70.73**	1123	1077	4.1
2.	62.08	**100**	25.00	**100**	248	178	28.4
3.	**98.35**	85.14	94.77	**95.59**	926	881	4.9
4.	97.37	**98.73**	66.67	**76.39**	240	236	1.6
5.	57.37	**97.00**	43.48	**63.77**	215	213	0.7
6.	56.10	**95.05**	61.01	**74.51**	422	414	2.0
7.	53.58	**98.57**	40.00	**80.00**	186	186	0
8.	80.95	**93.62**	**66.50**	58.67	343	332	3.2
9.	89.59	**96.52**	41.67	**87.50**	47	46	2.4
10.	93.49	**97.37**	44.44	**83.33**	397	384	3.1
11.	61.19	**99.87**	32.73	**94.55**	154	146	4.9
12.	72.90	**66.77**	77.42	**74.19**	167	159	3.8
Avg.	79.42	**92.62**	58.08	**79.66**	512	491	4.3

Now, a reasonable question is how much do we gain, in terms of ranking accuracy, and how much do we lose, in terms of total saved travel distance, if SVPL is used instead of STD for a real ridesharing data? Table 4.5 answers this question for our real database.

In Table 4.5, the first four columns show the accuracy of models with respect to two different metrics, namely nDCG and C-Index. The greater number of each measure has been emboldened for easier comparison. As expected, SVPL beats STD in ranking accuracy for most of the benchmarks. On the other hand, apart from the second benchmark, the loss rate of saved travel distance by using SVPL is less than 5%. The last row illustrates the weighted average of rows, where the weight for each row is proportional to the number of records for that benchmark (last column in Table 4.3); thus, a benchmark with a greater number of records has a proportionally greater impact on the average.

4.5.4 SVPL Versus the Worst Point Model

The results in Table 4.5 might be conclusive enough to lead us to believe that considering user preferences in ranking ride-matching items will increase the

Table 4.6: The ranking performance of SVPL and WPM are shown for two metrics, nDCG and C-Index.

Benchmark	nDCG		C-Index	
	WPM	SVPL	WPM	SVPL
1.	92.86	**94.81**	59.94	**70.73**
2.	80.12	**100**	62.50	**100**
3.	74.61	**85.14**	83.39	**95.59**
4.	90.90	**98.73**	45.14	**76.39**
5.	63.80	**97.00**	39.86	**63.77**
6.	94.60	**95.05**	73.01	**74.51**
7.	34.29	**98.57**	00.00	**80.00**
8.	**93.62**	**93.62**	**58.67**	**58.67**
9.	96.51	**96.52**	**87.50**	**87.50**
10.	94.41	**97.37**	55.55	**83.33**
11.	60.96	**99.87**	27.27	**94.55**
12.	**67.24**	66.77	35.48	**74.19**
Weighted Avg.	82.52	**92.62**	60.02	**79.66**

accuracy. However, what if an ad-hoc ranking strategy, which is generally sensible for all users, is adopted? In this situation, it is presumed all users have relatively similar preferences.

To examine this question, we assess SVPL against a somewhat simple-minded way of ranking which is called *Worst Point Model* (WPM). In WPM, a hypothetical matching trip with all features equal to 0, is assumed as the worst match. This assumption makes some sense, since it means that the worst matching is when the positional component, temporal component, *is favourable*, *has image* and so forth, all are 0. Afterwards, the score of each item is found from its Euclidean distance from the worst point; i.e., a higher score is produced by going further away from the worst point. Needless to say no learning process is involved in acquiring these scores. Therefore, a ranking of ridesharing opportunities, based on their distance from the worst point, will be suggested. Table 4.6 shows that SVPL produces a higher accuracy than WPM for most of benchmarks.

Table 4.7: The ranking accuracy comparison between SVPL and three regression methods, namely linear regression (LR), neural network regression (NNR) and support vector machine regression (SVM).

Bench.	nDCG				C-Index			
	LR	NNR	SVM	SVPL	LR	NNR	SVM	SVPL
1.	86.47	85.81	85.98	**94.81**	68.38	51.07	58.23	**70.73**
2.	100	100	100	100	100	100	100	100
3.	83.15	77.56	83.55	**85.14**	71.46	80.02	82.54	**95.59**
4.	98.54	**99.22**	93.65	98.73	74.31	**84.72**	48.61	76.39
5.	70.29	93.70	55.85	**97.00**	61.59	**72.46**	25.36	63.77
6.	70.12	69.88	67.37	**95.05**	67.01	67.01	53.52	**74.51**
7.	94.52	45.24	45.24	**98.57**	60.00	40.00	40.00	**80.00**
8.	**95.14**	91.54	83.55	93.62	65.17	57.67	56.83	**58.67**
9.	90.45	90.45	**96.52**	**96.52**	37.50	37.50	**87.50**	**87.50**
10.	92.11	94.74	94.74	**97.37**	50.00	66.67	66.67	**83.33**
11.	77.83	99.15	77.83	**99.87**	89.09	78.18	89.09	**94.55**
12.	61.52	61.04	66.49	**66.77**	48.39	29.03	70.97	**74.19**
Avg.	84.04	83.77	80.77	**92.62**	65.11	63.89	64.93	**79.66**

4.5.5 SVPL and Regression Models

Our experiments, thus far, have shown that learning the preferences of users individually is an auspicious direction in ride-matching. However, as explained in Section 4.3.1, using a regression-based method is another viable approach for learning weights. Recall that SVPL and the regression method require different formats for the input data, though they both eventually produce a utility function. For the regression method, the main user responses (class labels) are converted to scalar values, and for SVPL, a set of pairwise comparisons among ride-matching records is created (Section 4.3.1 illustrates how).

Thus, in this section, the performance of SVPL is compared with three regression methods in Table 4.7. These rival methods are *Linear Regression*, *Neural Network Regression* and *Support Vector Machine Regression* [NKNW96, CH15, Bot91, DBK+97] [5] which are respectively abbreviated as LR, NNR and SVM.

As seen in Table 4.7, SVPL outperforms the regression methods overall, which could evidently indicate that pairwise comparisons give a more natural representation of user preferences rather than simply converting labels to numeric

[5] We have used the scikit-learn library, that is available in http://scikit-learn.org/, for the implementation of these methods.

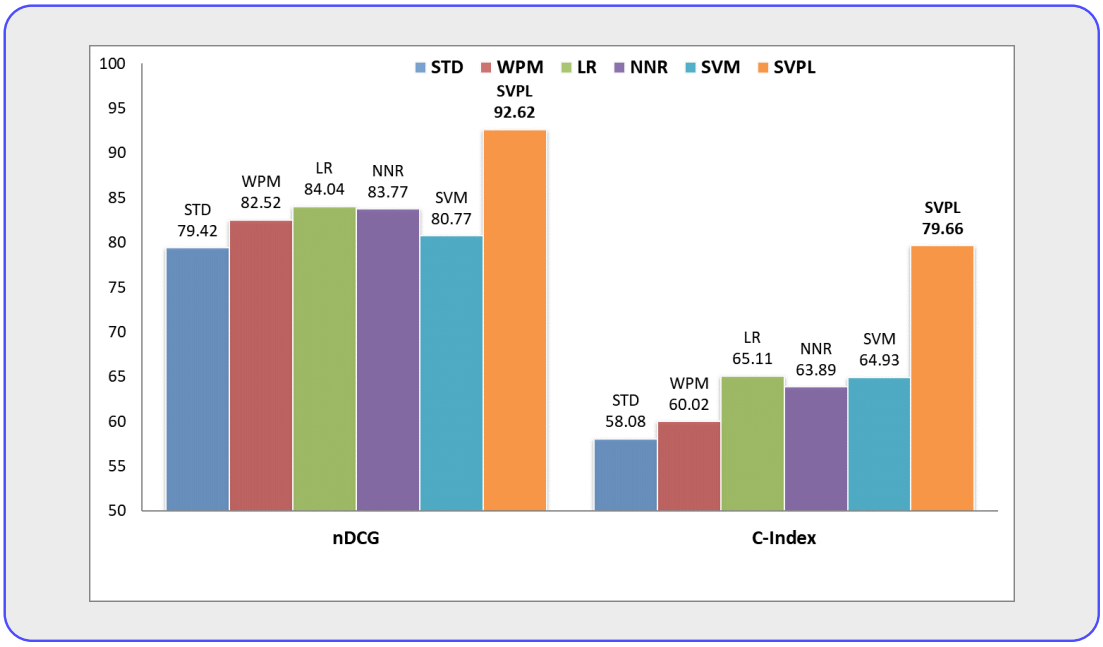

Figure 4.5: The overall accuracy of models according to nDCG and C-index metrics. The models compared are: ranking based on saved travel distance (STD), Worst Point Model (WPM), Linear Regression (LR), Neural Network Regression (NNR), Support Vector Machine Regression (SVM) and Support Vector Preferences Learner (SVPL).

values.

Finally, Figure 4.5 gives an overview of our experimental results, where the weighted average of the accuracy of each model (last columns in tables) have been drawn as a bar chart. The results show a clear superiority of SVPL over the other methods for the dataset.

4.6 Summary and Future Works

The advent of advanced technologies including GPS, web, and mobile technologies for real-time communication provides a unique opportunity to form new dynamic ridesharing systems, which could potentially provide substantial societal and environmental benefits. At the heart of the ridesharing concept, matching drivers and riders in real-time is prominently featured as a challenge.

In this section, we have presented novel aspects of the automatic ride-matching system, making use of user preferences. We believe that a good understanding of participant behaviour and preferences will be essential; if ride-share matches

do not satisfy participant preferences, the participant may not make use of the ride-share system in the future. Moreover, learning user preferences softens the traditional way of setting hard constraints, and removes the need for eliciting some complex parameters from participants.

Unlike the prevalent systems in ridesharing, our approach gives freedom to participants in the choice of ridesharing partners; it is supported by the fact that the user may not approve of the assigned matching found from the optimisation of the whole system. Of course, providing the flexibility in the matching process may cause deviation from the optimal solution with respect to the overall saved travel time. However, the first part of our experiments in Section 4.5.3 indicated that this loss could be quite small.

SVPL as a model with the ability to learn user preferences in a natural way, was discussed in Section 4.4. Our experimental results showed the effectiveness of SVPL, in comparison with five competitor strategies. The intention is that SVPL could contribute to the suggestion of a menu of good choices to the user in a ridesharing system.

A natural extension would be if the preferences set (pairwise comparisons) derived from the user's feedback, could be associated with some degree of uncertainty. For instance, if it is said that the user probably prefers the case A to B, with certainty degree of 0.7. Other interesting subjects related to this chapter that could be addressed in future works, include handling unknown values in the features space, and online learning.

Chapter 5

Scaling-Invariant Maximum Margin Preference Learning

5.1 Introduction

There is a growing trend towards personalisation for services in many real-world application domains, such as e-commerce, marketing, and entertainment. This involves capturing user preferences over alternative choices, e.g., products, movies and hotels. One may view this as an enhanced variation of supervised learning, known as *preference learning*, where instead of tagging an instance with a single label, preference relations are expressed over instances [YMH09, BGH10]. One natural way to express preferences over items is to represent them in the form of pairwise comparisons, stating that one alternative a is preferred over another one b, where an alternative is associated with a feature vector, i.e., a vector of values for a number of features.

As we discussed in the previous chapter, an established approach for modelling preferences makes use of the concept of a *utility function* that is learned from preference input pairs. Then, for a pair of test vectors (α, β), this function assigns an abstract degree of utility to each test vector, implying which test vector is preferred to which [FH10]. Support Vector Machine (SVM) approaches [Bur98] have inspired the development of several methods for learning the utility function, such as OrderSVM [KHM05], SVOR [HGO99] and SVMRank [Joa02].

In a method such as SVMRank, when the utility function has been learned, rescaling a pair of test vectors makes no difference to the result, i.e., α is preferred to β if and only if $r\alpha$ is preferred to $r\beta$ for any strictly positive scale factor r. The same does not hold for the input pairs: different ways of scaling preference input pairs may lead to a very different utility function being learned. However, it is arguable that in many contexts, a preference for a over b can be considered as conveying essentially equivalent information to a preference for ra over rb. For instance, knowing that the movie with the feature vector a is preferred to one with the feature vector b, we would often expect that $2a$ is preferred to $2b$. This suggests defining a more cautious preference relation by saying that a test vector α is *inputs-scaling-invariant preferred* to β if α is preferred to β for all choices of scalings of preference input pairs.

An analogous form of preference relation considers the scaling of features, where α is *features-scaling-invariant preferred* to β if α is preferred to β for all choices of scalings of features. Part of the motivation for this is that for any SVM-based method is necessary to scale (normalize) features beforehand.

This is because these methods are not invariant to the scale of their input feature spaces; for example, a particular feature with very large values, compared with the other features, might effectively veto the effect of other features on the objective function. Therefore, these methods are clearly sensitive to the way features are scaled [SKF08, BHW10]. The common practice for scaling is based on the properties of input instances [JD88, AH01, TD00]; as an example of a scaling method, the value of a feature is subtracted by the minimum of all values of that feature in the dataset and divided by the difference between the maximum and minimum. So, the scaling, and therefore the resulting preference relation, can sometimes depend strongly on precisely which preference inputs are received. There can thus be subjective, and even rather arbitrary, choices in the scaling of the feature spaces; different ways lead to different preference relations.

Taking into account both forms of rescaling mentioned above, we can also consider a still more cautious relation in which α is *scaling-invariant* preferred to β if it is preferred for all choices of scalings of features and preference input pairs simultaneously.

In this chapter, we define and analyse these more cautious preference relations that are invariant to the scaling of features, or preference inputs, or both simultaneously. The rest of the chapter is organised as follows. The next section introduces the terminology being used throughout the chapter and explains two preliminary preference relations. Section 5.3 considers the effect of rescaling of preference input pairs, and characterises a preference relation that is invariant to the scaling of preference input pairs. Similarly, the two other relations, where features are rescaled and where both features and preference inputs are rescaled simultaneously, are characterised in Sections 5.4 and 5.5 respectively. As the relations are based on the assumption that the input pairs are consistent, we briefly discuss three possible approaches to deal with inconsistencies in Section 5.6. The characterisations of relations lead to the computational methods in Section 5.7 for testing dominance with respect to the induced relations. In Section 5.8, we consider two kinds of optimality operator to choose a subset of alternatives as optimal solutions with regard to each preference approach. We report the experimental results that are carried out on derivatives of two real databases in Section 5.9; the experiments compare relations based on (a) the number of test pairs in which one dominates the other; (b) the number of optimal solutions found according to the defined optimality operators; and (c) the computation time. Section 5.10 concludes, with a discussion of potential

extensions. The appendix of this chapter includes the proofs of the lemmas, with the exception of some shorter proofs which are included in the main body of the chapter.

5.2 Preliminaries

In this section, we describe some notation and two preference relations that provide a basis for the following sections. Since there are inevitably many symbols and results to keep track of, Table 5.1 includes a glossary of symbols.

Recall the setting of the problem that we explained in Section 4.4 where the feature vector $a_i \in \mathbb{R}^n$ has been preferred over $b_i \in \mathbb{R}^n$, for each $i \in I = \{1, \ldots, m\}$. Also, the *preference inputs*, Λ, is $\{\lambda_i : i \in I\}$, where for each i, $\lambda_i = a_i - b_i$. By assuming a linear weighting model, each pair (a_i, b_i) expresses a linear restriction $a_i \cdot w > b_i \cdot w$ on an unknown weight vector $w \in \mathbb{R}^n$. [1] Thus, a *feasible* w satisfies $\lambda \cdot w > 0$ for all $\lambda \in \Lambda$.

Let us denote the feasible set by $\Lambda^>$ (=$\{w \in \mathbb{R}^n : \forall \lambda \in \Lambda, w \cdot \lambda > 0\}$), and associate the hyperplane $H_w = \{x \in \mathbb{R}^n : x \cdot w = 0\}$ with a feasible $w \in \Lambda^>$. We also assume that the preference inputs are consistent, so that $\Lambda^>$ is non-empty. Later, in Section 5.6, we discuss how to cope with the inconsistency in the data.

Example 13 » Representation of The Notation

Suppose that $n = 2$ and let the preference inputs Λ be $\{(2,1),(1,2),(1,1)\}$ (see Figure 5.1(a)). Then, a feasible $w \in \mathbb{R}^2$ satisfies these three conditions: (i) $2w(1)+w(2) > 0$, (ii) $w(1)+2w(2) > 0$ and (iii) $w(1)+w(2) > 0$. The feasible set, $\Lambda^>$, is shown in Figure 5.1(b) as the open space surrounded by dotted lines, i.e., both shaded regions. In Figure 5.1(a), the dotted line ($x + y = 0$) is a feasible hyperplane since it could be associated with a feasible point, such as $(\frac{1}{2}, \frac{1}{2})$.

[1] This linear weighting assumption is less restrictive than it sounds; for instance, we could form additional features representing e.g., pairwise products of the basic features, enabling a richer representation of the utility function.

5.2.1 Consistency Based Relation

One natural preference relation, $\succcurlyeq_{\Lambda}^{\mathrm{C}}$, which has been explored, for example, in [MRW13], is given as follows:

Definition 30: Consistency-based Preference Relation, $\succcurlyeq_{\Lambda}^{\mathrm{C}}$

The test vector α is *consistency-based preferred* to β, i.e., $\alpha \succcurlyeq_{\Lambda}^{\mathrm{C}} \beta$, if and only if $w \cdot \alpha \geq w \cdot \beta$ for all feasible $w \in \Lambda^{>}$. This means that dominance of α over β is consistent with the fact that for all $i \in I$, a_i has dominated b_i.

In addition to $\Lambda^{>}$, we define for any finite $\Lambda \subseteq I\!R^n$, the following two sets:

- $\Lambda^{\geq} = \{w \in I\!R^n : \forall \lambda \in \Lambda,\ w \cdot \lambda \geq 1\}$ ($\Lambda^{\geq}$ is the darkly shaded region in Figure 5.1(b)); and
- $\Lambda^{*} = \{w \in I\!R^n : \forall \lambda \in \Lambda,\ w \cdot \lambda \geq 0\}$ which is actually the dual cone of Λ (the closed space surrounded by dotted lines in Figure 5.1(b)).

Proposition 4 below states two other alternative ways to determine if $\alpha \succcurlyeq_{\Lambda}^{\mathrm{C}} \beta$ (just consider $\gamma = \alpha - \beta$). The proof uses Lemmas 2 and 3.

Lemma 2

Consider any $\Lambda \subseteq I\!R^n$. If $\Lambda^{>}$ is non-empty then Λ^{*} is the topological closure of $\Lambda^{>}$.

The following lemma is a well-known result for convex cones. Recall that the convex cone generated by Λ, $co(\Lambda)$, is the smallest convex cone containing Λ (this is the darkly shaded region in Figure 5.1(a)); i.e., the set of all vectors in $I\!R^n$ that can be written as $\sum_{\lambda \in \Lambda} r_\lambda \lambda$, where r_λ are arbitrary non-negative reals. Elements of $co(\Lambda)$ are said to be *positive linear combinations of elements of* Λ.

Lemma 3

Consider any finite $\Lambda \subseteq I\!R^n$ and any $u \in I\!R^n$. Then, $\Lambda^* \subseteq \{u\}^*$ if and only if $u \in co(\Lambda)$.

To illustrate, $u = (3,3)$ is in $co(\Lambda)$ in Figure 5.1, and $\{u\}^* = \{(x,y) : 3x + 3y \geq 0\}$ clearly contains $\Lambda^{\geq}$, the darkly shaded region in Figure 5.1(b).

Proposition 4: Determining the $\succcurlyeq_\Lambda^{\mathrm{C}}$ Relation

Consider any finite $\Lambda \subseteq I\!R^n$ such that $\Lambda^{>} \neq \emptyset$ and any $\gamma \in I\!R^n$. Then, the following conditions are equivalent. Thus, any of these are equivalent to $\gamma \succcurlyeq_\Lambda^{\mathrm{C}} \mathbf{0}$.

(i) for all $w \in \Lambda^{>}$, $w \cdot \gamma \geq 0$.

(ii) for all $w \in \Lambda^{\geq}$, $w \cdot \gamma \geq 0$.

(iii) $\gamma \in co(\Lambda)$.

Proof: (i) $\Rightarrow$ (ii): This follows immediately from $\Lambda^{\geq} \subset \Lambda^{>}$.

(i) $\Leftarrow$ (ii): Suppose that for all $w \in \Lambda^{\geq}$, $w \cdot \gamma \geq 0$, and consider any $u \in \Lambda^{>}$. Let $a_u = \min_{\lambda \in \Lambda} u \cdot \lambda$ which is clearly greater than zero, and let $u' = \frac{u}{a_u}$. For any $\lambda \in \Lambda$, $u \cdot \lambda \geq a_u$ which implies that $u' \cdot \lambda \geq 1$, and thus, $u' \in \Lambda^{\geq}$. Because $u' \cdot \gamma \geq 0$, we have also, $u \cdot \gamma \geq 0$.

(i) $\Leftrightarrow$ (iii): (i) means $\Lambda^{>} \subseteq \{\gamma\}^*$ which, because $\{\gamma\}^*$ is a closed set, holds if and only if $Cl(\Lambda^{>}) \subseteq \{\gamma\}^*$, i.e., $\Lambda^* \subseteq \{\gamma\}^*$, using Lemma 2. Lemma 3 implies that this is if and only if (iii) $\gamma \in co(\Lambda)$. ■

Table 5.1: The glossary of symbols being used throughout the chapter.

Symbol	Meaning
n	number of features.
m	number of preference input pairs.
(a_i, b_i)	a preference input pair when a_i has been preferred to b_i.
I	defined as $\{1, \ldots, m\}$; i.e., the index set for input pairs.
$\cdot$	the dot product, e.g., $(2,3) \cdot (3,1) = 9$.
Λ	is the (finite) set of preference inputs; i.e., $\{\lambda_i : \forall i \in I, \lambda_i = a_i - b_i\}$; e.g., the black points marked in Figure 5.1(a).
$\Lambda^{>}$	defined as $\{w \in I\!R^n : \forall \lambda \in \Lambda,\ w \cdot \lambda > 0\}$; e.g., the open space surrounded by dotted lines in Figure 5.1(b).
$\Lambda^{\geq}$	defined as $\{w \in I\!R^n : \forall \lambda \in \Lambda,\ w \cdot \lambda \geq 1\}$; e.g., the darkly shaded region in Figure 5.1(b).
Λ^{*}	defined as $\{w \in I\!R^n : \forall \lambda \in \Lambda,\ w \cdot \lambda \geq 0\}$; e.g., the closed space surrounded by dotted lines in Figure 5.1(b).
$co(\Lambda)$	the smallest convex cone containing Λ; e.g., the darkly shaded region in Figure 5.1(a).
$\succcurlyeq_{\Lambda}^{\mathrm{C}}$	the consistency based relation; i.e., $\alpha \succcurlyeq_{\Lambda}^{\mathrm{C}} \beta$ iff for all $w \in \Lambda^{>}$, $w \cdot (\alpha - \beta) \geq 0$.
$S \setminus \{x\}$	from the set S, the element x is excluded.
$\Lambda^{*}+\{u\}$	for the vector $u \in I\!R^n$, defined as $\{w + u : w \in \Lambda^{*}\}$.
$\|w\|$	Euclidean norm of w.
$\omega_{\Lambda}^{\star}$	the element with minimal norm in $\Lambda^{\geq}$; e.g., $(\frac{1}{2}, \frac{1}{2})$ in Figure 5.1(b).
$\succcurlyeq_{\Lambda}^{mm}$	the maximum margin preference relation; i.e., $\alpha \succcurlyeq_{\Lambda}^{mm} \beta$ iff $\alpha \cdot \omega_{\Lambda}^{\star} \geq \beta \cdot \omega_{\Lambda}^{\star}$.
$I\!R_{+}$	the set of strictly positive reals.
$I\!R_{+}^{m}$	the set of vectors $u \in I\!R^m$ with each component strictly positive.
$\mathbf{t}$	$\in I\!R_{+}^{m}$, the rescaling vector for preference inputs.
$\Lambda_{\mathbf{t}}$	defined as $\{\mathbf{t}(i)\lambda_i : \forall i \in I\}$, i.e., preference inputs being rescaled by $\mathbf{t}$.
$\succcurlyeq_{\Lambda}^{\mathrm{I}}$	the relation that is invariant to the rescaling of inputs; i.e., $\alpha \succcurlyeq_{\Lambda}^{\mathrm{I}} \beta$ iff for all $\mathbf{t} \in I\!R_{+}^{m}$, $\alpha \succcurlyeq_{\Lambda_{\mathbf{t}}}^{mm} \beta$.
$\mathrm{SI}(\Lambda)$	defined as $\{\omega_{\Lambda_{\mathbf{t}}}^{\star} : \mathbf{t} \in (0,1]^m\}$; e.g., the intersection of darkly shaded regions in sub-figures 5.1(a) and (b).
τ	$\in I\!R_{+}^{n}$, the rescaling vector for features.
τ^{-1}	$\in I\!R_{+}^{n}$, given by $\tau^{-1}(j) = 1/\tau(j)$ for all $j \in \{1, \ldots, n\}$.
$\odot$	pointwise multiplication, e.g., $(2,3) \odot (3,1) = (6,3)$.
$\Lambda \odot \tau$	defined as $\{\lambda \odot \tau : \forall \lambda \in \Lambda\}$, i.e., features being rescaled by τ.
$\succcurlyeq_{\Lambda}^{\mathrm{F}}$	the relation that is invariant to the rescaling of features; i.e., $\alpha \succcurlyeq_{\Lambda}^{\mathrm{F}} \beta$ iff for all $\tau \in I\!R_{+}^{n}$, $\alpha \odot \tau \succcurlyeq_{\Lambda \odot \tau}^{mm} \beta \odot \tau$.
$\mathrm{SF}(\Lambda)$	defined as $\{\omega_{\Lambda \odot \tau}^{\star} \odot \tau : \tau \in I\!R_{+}^{n}\}$; e.g., the part of the line segment $x + y = 1$ strictly within the first quadrant in Figure 5.1(b).
$\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$	the relation that is invariant to the rescaling of features and inputs simultaneously.
$\mathrm{SIF}(\Lambda)$	defined as $\{\omega_{\Lambda_{\mathbf{t}} \odot \tau}^{\star} \odot \tau : \mathbf{t} \in (0,1]^m, \tau \in I\!R_{+}^{n}\}$; e.g., the part of the darkly shaded region that is strictly within the first quadrant (so not including the axes) in Figure 5.1(b).

5.2.2 Maximum Margin Preference Relation

Based on the principal idea in conventional SVM, SVMRank [Joa02] picks a single w from the feasible set that maximises the margin (leading to a stronger ordering than $\succcurlyeq_\Lambda^{\mathrm{C}}$); by *margin* we mean the perpendicular distance between the hyperplane H_w and the closest element of Λ to H_w. In simple terms, maximising the margin means choosing a feasible hyperplane that is as far as possible from Λ. One might view the distance between H_w and λ (i.e., $\frac{w\cdot\lambda}{\|w\|}$) as the degree to which w satisfies the preference λ, with the best w being those that satisfy each λ to the greatest degree. As we have seen in Section 4.4 and will prove formally in Theorem 5 below, the hyperplane that produces the maximum margin is equal to the hyperplane H_w where w uniquely has the minimum (Euclidean) norm in $\Lambda^{\geq}$. We denote the unique element of $\Lambda^{\geq}$ with the minimum norm by $\omega_\Lambda^\star$. In Figure 5.1(b), $(\frac{1}{2},\frac{1}{2})$ has uniquely minimal norm in $\Lambda^{\geq}$, so $\omega_\Lambda^\star = (\frac{1}{2},\frac{1}{2})$, and thus, the associated hyperplane with that point, $x+y=0$ in Figure 5.1(a), has the maximum margin.

Theorem 5: Equality of Maximising Margin and Minimising Norm

Let Λ be a finite subset of $\mathbb{R}^n$, and suppose that $\Lambda^{>}$ is non-empty. Then the following all hold.

(i) $\Lambda^{\geq}$ is non-empty;

(ii) there exists a unique element $\omega_\Lambda^\star$ in $\Lambda^{\geq}$ with minimum norm;

(iii) w maximises margin (i.e., $d_w = \min_{\lambda\in\Lambda}\frac{w\cdot\lambda}{\|w\|}$) within $\Lambda^{>}$ if and only if w is a strictly positive scalar multiple of $\omega_\Lambda^\star$, i.e., there exists $r\in\mathbb{R}$ with $r>0$ such that $w = r\omega_\Lambda^\star$.

Proof: If $\Lambda^{>}$ is non-empty then, by Lemma 1(i), $\bar{w}\in\Lambda^{\geq}$ for any $w\in\Lambda^{>}$. Thus, $\Lambda^{\geq}$ is non-empty, showing (i). Regarding (ii), since $\Lambda^{\geq}$ is convex and topologically closed, there exists a unique element $\omega_\Lambda^\star$ in $\Lambda^{\geq}$ with minimum norm, by a standard result (for a proof, see e.g., Proposition 4 of [WM16b]).

To prove (iii), consider any $w \in \Lambda^{>}$. As we just showed, $\bar{w} \in \Lambda^{\geq}$, so minimality of $\omega^\star_\Lambda$ implies that $\|\bar{w}\| \geq \|\omega^\star_\Lambda\|$ which equals $\|\overline{\omega^\star_\Lambda}\|$, using Lemma 1(ii). Lemma 1(iv) then implies that $d_w \leq d_{\omega^\star_\Lambda}$, which implies that $\omega^\star_\Lambda$ maximises d_w for all $w \in \Lambda^{>}$. Also, if $d_w = d_{\omega^\star_\Lambda}$ then $\|\bar{w}\| = \|\omega^\star_\Lambda\|$, and thus, $\bar{w} = \omega^\star_\Lambda$ by uniqueness of $\omega^\star_\Lambda$. Then $\frac{w}{a_w} = \omega^\star_\Lambda$ so w is a positive scalar multiple of $\omega^\star_\Lambda$. Finally, for any $r > 0$, $d_{r\omega^\star_\Lambda} = d_{\omega^\star_\Lambda}$ so $r\omega^\star_\Lambda$ maximises d_w for all $w \in \Lambda^{>}$. ■

More general versions of this result that allow additional linear restrictions on the feasible set $\Lambda^{>}$ are given in [WM16a, WM16b].

Definition 31: Max-margin Preference Relation, $\succcurlyeq^{mm}_\Lambda$

We define relation $\succcurlyeq^{mm}_\Lambda$ by, for $\alpha, \beta \in I\!R^n$, α is *max-margin-preferred* to β with respect to Λ (i.e., $\alpha \succcurlyeq^{mm}_\Lambda \beta$) if and only if $\alpha \cdot \omega^\star_\Lambda \geq \beta \cdot \omega^\star_\Lambda$, where $\omega^\star_\Lambda$ has uniquely minimal norm in $\Lambda^{\geq}$.

The relation $\succcurlyeq^{mm}_\Lambda$ is a total pre-order, since it is transitive and for any $\alpha, \beta \in I\!R^n$ we have $\alpha \succcurlyeq^{mm}_\Lambda \beta$ or $\beta \succcurlyeq^{mm}_\Lambda \alpha$ (or both). Considering Λ as in Example 13, $\omega^\star_\Lambda = (1/2, 1/2)$, and $\{(1/2, 1/2)\}^* = \{(x, y) : x + y \geq 0\}$; so, for all γ in the positive half-space of $x + y = 0$, $\gamma \succcurlyeq^{mm}_\Lambda \mathbf{0}$.

5.3 Rescaling of Preference Inputs

As discussed in the introduction of this chapter, it seems natural that a preference relation should not depend on how the preference inputs are scaled. As shown below in Example 14, this does not hold for the max-margin preference relation. In this section we define and give a characterisation of a preference relation $\succcurlyeq^{\mathrm{I}}_\Lambda$ that is invariant to rescaling of the preference inputs Λ.

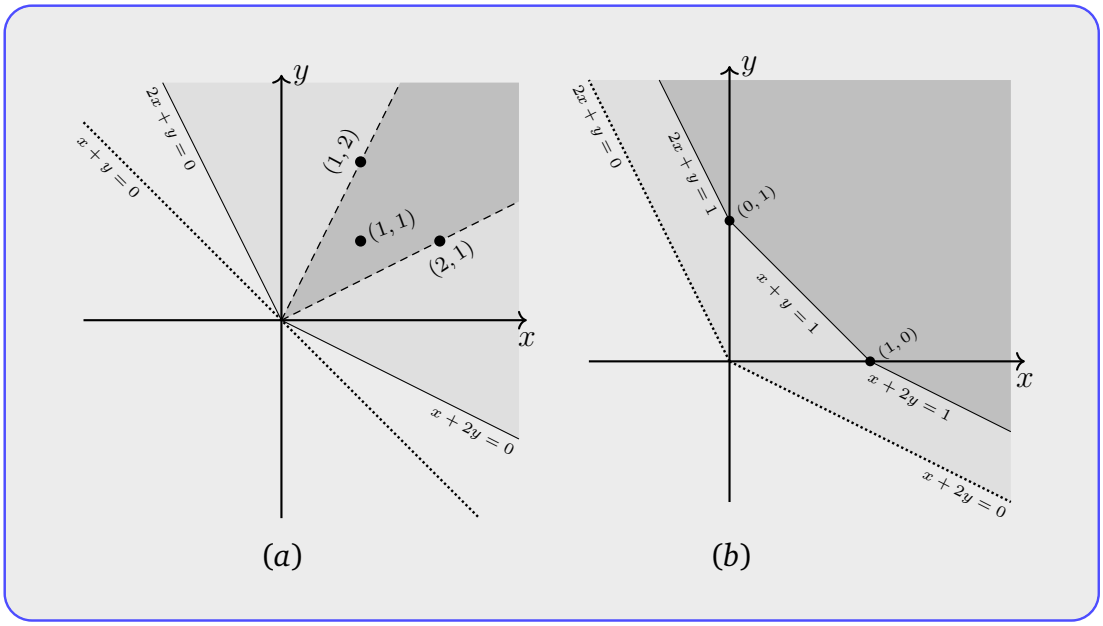

Figure 5.1: **(a)** The visual representation of Example 13 when $\Lambda = \{(2,1),(1,2),(1,1)\}$. If the element γ is in (i) the darkly shaded region; (ii) the first quadrant; (iii) all the shaded region; and (iv) the positive half space of $x + y = 0$ then γ will dominate $\mathbf{0}$ under relation (i) $\succcurlyeq_{\Lambda}^{\mathrm{C}}$; (ii) $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ and $\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$ (they lead to the same result in this example); (iii) $\succcurlyeq_{\Lambda}^{\mathrm{I}}$; and (iv) $\succcurlyeq_{\Lambda}^{mm}$, respectively. **(b)** $\Lambda^{\geq}$ is the darkly shaded region, $\mathrm{SIF}(\Lambda)$ is the part of $\Lambda^{\geq}$ that is strictly within the first quadrant (so not including the axes), $\mathrm{SF}(\Lambda)$ is the part of the line segment $x + y = 1$ strictly within the first quadrant, and $\mathrm{SI}(\Lambda)$ is the intersection of $\Lambda^{\geq}$ and $co(\Lambda)$ (i.e., the intersection of darkly shaded regions in both sub-figures).

5.3.1 Defining Inputs-Rescaling-Invariant Relation

Consider the effect of rescaling the preference inputs Λ by $\mathbf{t} \in \mathbb{R}_{+}^{m}$ (where $\mathbb{R}_{+}^{m}$ is the set of strictly positive reals in m-dimensional), with each preference input being multiplied by a strictly positive scalar, so that the rescaled preference input set is defined as $\Lambda_{\mathbf{t}} = \{\mathbf{t}(i)\lambda_i : i \in I\}$. We then have $(\Lambda_{\mathbf{t}})^{\geq} = \Lambda_{\mathbf{t}}^{\geq} = \{w \in \mathbb{R}^n : \forall i \in I,\ w \cdot (\mathbf{t}(i)\lambda_i) \geq 1\}$. We will write $\mathbf{t}(i)$ as $\mathbf{t}_i$ for brevity. Let us say that α *is max-margin-preferred to* β *under rescaling* $\mathbf{t}$ if $\alpha \succcurlyeq_{\Lambda_t}^{mm} \beta$. Now, it can easily happen that α is preferred to β under one rescaling, but not under another.

Example 14 » The Effect of Scaling of Preference Inputs

Consider $\mathbf{t} = (3/5, 1/5, 1)$ *rescaling* Λ *in Example 13. Then,* $\Lambda_{\mathbf{t}}$ *equals*

$\{(6/5, 3/5), (1/5, 2/5), (1, 1)\}$. *In Figure 5.2(a),* $\Lambda_{\mathbf{t}}^{\geq}$ *is the whole shaded region, and it can be seen that* $\omega_{\Lambda_{\mathbf{t}}}^{\star} = (1, 2)$ *which means the hyperplane with the maximum margin for* $\Lambda_{\mathbf{t}}$ *is* $x + 2y = 0$ *(instead of* $x + y = 0$*). Then,* $(2, -1.5) \succcurlyeq_{\Lambda}^{mm} (0, 0)$ *because* $(2, -1.5) \cdot (1, 1) = 0.5 > 0$*, whereas* $(2, -1.5) \not\succcurlyeq_{\Lambda_t}^{mm} (0, 0)$ *because* $(2, -1.5) \cdot (1, 2) = -1 < 0$.

However, it seems natural to assume that if the user prefers a_i over b_i then he will also prefer $\mathbf{t}_i a_i$ over $\mathbf{t}_i b_i$ for any $\mathbf{t}_i \in I\!R_+$. Also, for test vectors α and β, if $\alpha \succcurlyeq_{\Lambda}^{mm} \beta$ then, for any positive real r, we have $r\alpha \succcurlyeq_{\Lambda}^{mm} r\beta$; since the resultant preferences are invariant to such rescaling, it seems reasonable that the same would hold for the input preferences.

We therefore consider a more robust relation, which is invariant to the scaling of the preference inputs, with α being *inputs-scaling-invariant* preferred to β only if it is max-margin preferred for all rescalings $\mathbf{t} \in I\!R_+^m$ of the preference inputs.

Definition 32: Inputs-scaling-invariant Preference Relation, $\succcurlyeq_{\Lambda}^{\mathrm{I}}$

We define relation $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ by, for $\alpha, \beta \in I\!R^n$, $\alpha \succcurlyeq_{\Lambda}^{\mathrm{I}} \beta$ if and only if α is max-margin-preferred to β over all rescalings of preference inputs, i.e., if for all $\mathbf{t} \in I\!R_+^m$, $\alpha \succcurlyeq_{\Lambda_{\mathbf{t}}}^{mm} \beta$.

So far, we have assumed that each component $\mathbf{t}_i$ of $\mathbf{t}$ can be any strictly positive scalar. However, in Proposition 8 below, we will show that if each $\mathbf{t}_i$ is restricted to be in $(0, 1]$, the result for relation $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ will not change. This is not surprising, since, e.g., doubling each component of $\mathbf{t}$ will not change the relation $\succcurlyeq_{\Lambda_{\mathbf{t}}}^{mm}$. This simplification will be helpful in the computation of the $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ relation.

The following lemma and proposition are used to prove Proposition 8.

Lemma 6

Consider any finite $\Lambda \subseteq I\!R^n$, any $\mathbf{t} \in I\!R_+^m$, any $r \in I\!R_+$, and any $v \in I\!R^n$. If

$\mathbf{t}' = \frac{t}{r}$ then the following results hold.

(i) $v \in \Lambda^{\geq}_{\mathbf{t}}$ if and only if rv is in $\Lambda^{\geq}_{\mathbf{t}'}$.

(ii) $\omega^{\star}_{\Lambda_{\mathbf{t}'}} = r\omega^{\star}_{\Lambda_{\mathbf{t}}}$; i.e., v has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}}$ if and only if rv has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}'}$.

To illustrate, consider $\mathbf{t} = (3, 1, 5)$ and $r = 5$. So, $\mathbf{t}' = (3/5, 1/5, 1)$. We know from Example 14 that $\omega^{\star}_{\Lambda_{\mathbf{t}'}} = (1, 2)$ and $\Lambda^{\geq}_{\mathbf{t}'}$ is the intersection of $6x + 3y \geq 5$ and $x + 2y \geq 5$ (i.e., all the shaded region in Figure 5.2(a)). It can be shown similarly that $\Lambda^{\geq}_{\mathbf{t}}$ is the intersection of $6x + 3y \geq 1$ and $x + 2y \geq 1$, leading to $\omega^{\star}_{\Lambda_{\mathbf{t}}} = (1/5, 2/5) = \frac{1}{r}\omega^{\star}_{\Lambda'_{\mathbf{t}}}$.

Proposition 7

Consider any finite $\Lambda \subseteq I\!R^n$, any $\mathbf{t} \in I\!R^m_+$, and any $r \in I\!R_+$. Then, if $\mathbf{t}' = \frac{t}{r}$ then $\succcurlyeq^{mm}_{\Lambda_{\mathbf{t}'}}$ is equal to $\succcurlyeq^{mm}_{\Lambda_{\mathbf{t}}}$; i.e., for any α and $\beta \in I\!R^n$, $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}'}} \beta$ if and only if $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}}} \beta$.

Proof: $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}'}} \beta$ if and only if $\omega^{\star}_{\Lambda_{\mathbf{t}'}} \cdot \alpha \geq \omega^{\star}_{\Lambda_{\mathbf{t}'}} \cdot \beta$ which, by Lemma 6 (ii), holds if and only if $r\omega^{\star}_{\Lambda_{\mathbf{t}}} \cdot \alpha \geq r\omega^{\star}_{\Lambda_{\mathbf{t}}} \cdot \beta$ which is clearly if and only if $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}}} \beta$. ■

Proposition 8: Restricting The Scaling Factor $\mathbf{t}$ in $(0, 1]^m$

Consider any finite $\Lambda \subseteq I\!R^n$ and any $\alpha, \beta \in I\!R^n$. Then, $\alpha \succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ if and only if for all $\mathbf{t} \in (0, 1]^m$, $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}}} \beta$.

Proof: $\alpha \succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ iff, by definition of $\succcurlyeq^{\mathrm{I}}_{\Lambda}$, for all $\mathbf{t} \in I\!R^m_+$, $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}}} \beta$, which, by Proposition 7, is if and only if for all $\mathbf{t} \in I\!R^m_+$, and any $r \in I\!R_+$, $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}'}} \beta$,

where $\mathbf{t}' = \frac{t}{r}$. In particular when $r = \max_{i \in I} \mathbf{t}_i$, this holds if and only if for all $\mathbf{t}' \in (0,1]^m$, $\alpha \succcurlyeq^{mm}_{\Lambda_{\mathbf{t}'}} \beta$. ■

Now, let us define $\mathrm{SI}(\Lambda)$ to be the set consisting solely of $\omega^\star_{\Lambda_\mathbf{t}}$ for all scalings $\mathbf{t} \in (0,1]^m$; i.e., $\mathrm{SI}(\Lambda) = \{\omega^\star_{\Lambda_\mathbf{t}} : \mathbf{t} \in (0,1]^m\}$. Thus, $u \in \mathrm{SI}(\Lambda)$ if and only if there exists $\mathbf{t} \in (0,1]^m$ such that u has minimal norm in $\Lambda^{\geq}_\mathbf{t}$. Proposition 8, along with Definition 31, immediately implies the following result.

Proposition 9

For preference inputs $\Lambda \subseteq I\!R^n$ and $\alpha, \beta \in I\!R^n$, we have $\alpha \succcurlyeq^{\mathrm{I}}_\Lambda \beta \iff$ for all $w \in \mathrm{SI}(\Lambda)$, $\alpha \cdot w \geq \beta \cdot w$.

For example, it can be shown that $\mathrm{SI}(\Lambda)$ in Figure 5.1 is the intersection of the darkly shaded regions in sub-figures (a) and (b) (see Theorem 16 below). Then, it can be seen that $(\mathrm{SI}(\Lambda))^*$ is all the shaded region in Figure 5.1(a). This implies that $\gamma \succcurlyeq^{\mathrm{I}}_\Lambda \mathbf{0}$ if and only if γ is in all the shaded region in Figure 5.1(a).

5.3.2 Characterisation of $\mathrm{SI}(\Lambda)$

Here, we mathematically characterise $\mathrm{SI}(\Lambda)$; this will lead to a computational method for the $\succcurlyeq^{\mathrm{I}}_\Lambda$ relation. Proposition 13 below implies that $\mathrm{SI}(\Lambda) \subseteq \Lambda^{\geq}$ and for every element u in $\mathrm{SI}(\Lambda)$, vector u has minimum norm in $\Lambda^* + \{u\} = \{w + u : w \in \Lambda^*\}$ (which equals $\{w' \in I\!R^n : \forall \lambda \in \Lambda, w' \cdot \lambda \geq u \cdot \lambda\}$). The proof uses the three lemmas below.

In the running example, assume $u = (1,2)$. Then, $\Lambda^* + \{u\} = \{(x,y) : 2x + y \geq 4, x + 2y \geq 5, x + y \geq 3\}$ which is the darkly shaded region in Figure 5.2(a) (place the origin on u and then draw Λ^*).

Lemma 10

Consider any finite $\Lambda \subseteq I\!R^n$, and any $\mathbf{t} \in (0,1]^m$. Then, for any $u \in \Lambda^{\geq}_\mathbf{t}$ we

have $\Lambda^* + \{u\} \subseteq \Lambda^{\geq}_{\mathbf{t}}$.

To illustrate, consider $\mathbf{t} = (3/5, 1/5, 1)$ as in Example 14, and choose $u = (1, 2)$. Then, $\Lambda^* + \{u\}$ is the darkly shaded region in Figure 5.2(a). We can see in the figure that $\Lambda^* + \{u\} \subseteq \Lambda^{\geq}_{\mathbf{t}}$ where $\Lambda^{\geq}_{\mathbf{t}}$ is all the shaded region.

Proof: For any $u \in I\!R^n$ and any $v \in \Lambda^* + \{u\}$, we have, by the definition of Λ^*, $\forall i \in I,\ (v - u) \cdot \lambda_i \geq 0$, which means that $v \cdot \lambda_i \geq u \cdot \lambda_i$. Also, since it is assumed that $u \in \Lambda^{\geq}_{\mathbf{t}}$, we have $\forall i \in I,\ u \cdot \lambda_i \geq 1/\mathbf{t}_i$. Thus, $\forall i \in I,\ v \cdot \lambda_i \geq 1/\mathbf{t}_i$, and so, $v \in \Lambda^{\geq}_{\mathbf{t}}$. ■

Lemma 11

Consider any finite $\Lambda \subseteq I\!R^n$, and any $u \in \Lambda^{\geq}$. Then, there exists $\mathbf{t} \in (0, 1]^m$ such that $\Lambda^{\geq}_{\mathbf{t}} = \Lambda^* + \{u\}$.

To illustrate, consider $u = (1, 2)$ and $\mathbf{t} = (1/4, 1/5, 1/3)$. Then, $\Lambda^{\geq}_{\mathbf{t}} = \{(x, y) : \frac{2}{4}x + \frac{1}{4}y \geq 1, \frac{1}{5}x + \frac{2}{5}y \geq 1, \frac{1}{3}x + \frac{1}{3}y \geq 1\}$ which is equal to $\Lambda^* + \{u\}$, the darkly shaded region in Figure 5.2(a).

Proof: $u \in \Lambda^{\geq}$ means that for all $i \in I$, $u \cdot \lambda_i \geq 1$, which implies that $0 < \frac{1}{u \cdot \lambda_i} \leq 1$. For all $i \in I$, let $\mathbf{t}_i = \frac{1}{u \cdot \lambda_i}$, and so $\mathbf{t} \in (0, 1]^m$. By definition, $w \in \Lambda^{\geq}_{\mathbf{t}}$ if and only if for all $i \in I$, $w \cdot \mathbf{t}_i \lambda_i \geq 1$. Now, $w \cdot \mathbf{t}_i \lambda_i \geq 1$ holds if and only if $w \cdot \lambda_i \geq u \cdot \lambda_i$, which is if and only if $(w - u) \cdot \lambda_i \geq 0$. Thus, $\Lambda^{\geq}_{\mathbf{t}} = \{w \in I\!R^n : \forall i \in I,\ (w - u) \cdot \lambda_i \geq 0\}$, which equals $\Lambda^* + \{u\}$. ■

Lemma 12

Consider any finite $\Lambda \subseteq I\!R^n$, and any $\mathbf{t} \in (0, 1]^m$. Then, $\Lambda^{\geq}_{\mathbf{t}} \subseteq \Lambda^{\geq}$.

Proof: Consider any $u \in \Lambda^{\geq}_{\mathbf{t}}$. Then for all $i \in I$, $u \cdot \lambda_i \geq \frac{1}{\mathbf{t}_i}$. Since each $\mathbf{t}_i$ is in $(0, 1]$ we have for all $i \in I$, $u \cdot \lambda_i \geq 1$, and thus, $u \in \Lambda^{\geq}$. ■

Proposition 13

Consider any $u \in I\!R^n$. Then, $u \in \mathrm{SI}(\Lambda)$ if and only if $u \in \Lambda^{\geq}$ and u has minimum norm in $\Lambda^* + \{u\}$. Thus, in particular, $\mathrm{SI}(\Lambda) \subseteq \Lambda^{\geq}$.

We know that $u = (1, 2) \in \mathrm{SI}(\Lambda)$ because it has minimum norm in $\Lambda_{\mathbf{t}}$ for $\mathbf{t} = (3/5, 1/5, 1)$. We can easily see that $u \in \Lambda^{\geq}$ and has minimum norm in $\Lambda^* + \{u\}$. Now, let v be any point between two black circles in Figure 5.2(a). Then, v does not have minimal norm in $\Lambda^* + \{v\}$; in fact, $(1, 2)$ minimises norm in $\Lambda^* + \{v\}$. We will see that $v \notin \mathrm{SI}(\Lambda)$.

Proof: $\Rightarrow$: $u \in \mathrm{SI}(\Lambda)$ means that there exists $\mathbf{t} \in (0, 1]^m$ such that $u \in \Lambda^{\geq}_{\mathbf{t}}$ and u has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}}$, which, since $\Lambda^{\geq}_{\mathbf{t}} \subseteq \Lambda^{\geq}$ by Lemma 12, implies that $u \in \Lambda^{\geq}$. Now, u also has the minimum norm in $\Lambda^* + \{u\}$ because firstly, $\Lambda^* + \{u\} \subseteq \Lambda^{\geq}_{\mathbf{t}}$ from Lemma 10, and secondly, u is clearly in $\Lambda^* + \{u\}$ since $\mathbf{0} \in \Lambda^*$.

$\Leftarrow$: Assume now that $u \in \Lambda^{\geq}$ and u has the minimum norm in $\Lambda^* + \{u\}$. By Lemma 11, there exists $\mathbf{t} \in (0, 1]^m$ such that u has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}} (= \Lambda^* + \{u\})$, and clearly $u \in \Lambda^{\geq}_{\mathbf{t}}$. Thus, $u \in \mathrm{SI}(\Lambda)$. ■

We will prove (in Proposition 15) that $co(\Lambda)$ is precisely the set of elements $u \in I\!R^n$ such that u has minimum norm in $\Lambda^* + \{u\}$. Together with Proposition 13, this will imply Theorem 16 below, which characterises SI. The following lemma is used in the proof; it states a basic property of a minimal norm element in a convex set.

Lemma 14

Consider any $u \in G$ where $G \subseteq I\!R^n$ is a convex set. Then, u has the minimum norm in G if and only if for all $v \in G$, $u \cdot (v - u) \geq 0$.

Proposition 15

Consider any finite $\Lambda \subseteq I\!R^n$ and any $u \in I\!R^n$. Then, u has minimum norm in $\Lambda^* + \{u\}$ if and only if $u \in co(\Lambda)$.

Proof: Clearly, $\Lambda^* + \{u\}$ is a convex set. Lemma 14 implies that u has minimum norm in $\Lambda^* + \{u\}$ if and only if for all $v \in \Lambda^* + \{u\}$, $u \cdot (v - u) \geq 0$. By writing $y = v - u$, this is if and only if for all $y \in \Lambda^*$, $u \cdot y \geq 0$, which holds if and only if for all $y \in \Lambda^*$, $y \in \{u\}^*$. Thus, u has minimum norm in $\Lambda^* + \{u\}$ if and only if $\Lambda^* \subseteq \{u\}^*$. Lemma 3 then implies the result. ■

Propositions 13 and 15 immediately imply the following theorem.

Theorem 16: $\mathrm{SI}(\Lambda)$ **Characterisation**

Consider any finite $\Lambda \subseteq I\!R^n$, any $u \in I\!R^n$. Then, $\mathrm{SI}(\Lambda) = co(\Lambda) \cap \Lambda^{\geq}$.

Theorem 16 shows that $\mathrm{SI}(\Lambda)$ in Figure 5.1 is the intersection of the darkly shaded regions in sub-figures (a) and (b).

Proof: u is in $\mathrm{SI}(\Lambda)$ if and only if by Proposition 13, $u \in \Lambda^{\geq}$ and u has minimum norm in $\Lambda^* + \{u\}$, which, from Proposition 15, holds if and only if $u \in \Lambda^{\geq}$ and $u \in co(\Lambda)$. ■

The following result leads immediately to an algorithm to determine, for arbi-

trary $\alpha, \beta \in I\!R^n$ if $\alpha \succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$, using a linear programming solver.

Corollary 17: The $\succcurlyeq^{\mathrm{I}}_{\Lambda}$ Relation Computation Method

For finite set $\Lambda \subset I\!R^n$, let $\lambda_i \in \Lambda$ be the i^{th} element of Λ where $i \in I = \{1, \ldots, |\Lambda|\}$. Consider any $u \in I\!R^n$. Then, u is in $\mathrm{SI}(\Lambda)$ if and only if for all $i \in I$, $u \cdot \lambda_i \geq 1$ and there exist non-negative reals r_i for each $i \in I$ such that $u = \sum_{i \in I} r_i \lambda_i$.

Proof: The result follows easily from Theorem 16 and the definition of $co(\Lambda)$ and $\Lambda^{\geq}$. ■

Now, we have the following procedure to compute the $\succcurlyeq^{\mathrm{I}}_{\Lambda}$ relation; Proposition 9 implies that for $\alpha, \beta \in I\!R^n$, $\alpha \not\succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ if and only if there exists $u \in \mathrm{SI}(\Lambda)$ such that $\alpha \cdot u < \beta \cdot u$, which, by Corollary 17, is if and only if there exists $u \in I\!R^n$ such that (i) $u \cdot (\beta - \alpha) > 0$, (ii) for all $i \in I$, $u \cdot \lambda_i \geq 1$, and (iii) there exist non-negative reals r_i for each $i \in I$ such that $u = \sum_{i \in I} r_i \lambda_i$.

5.4 Rescaling of Features

As discussed in the introduction, an important, and potentially problematic, pre-processing step in SVM methods is rescaling of the domain of each feature. In this section we define a preference relation $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ (based on preference inputs Λ) that is invariant to the relative scalings of the feature domains.

Normalization of features is a necessary phase in any SVM-based method. This task often involves translations and rescalings on the domain of each feature. It is evident that the maximum margin relation is unaffected by translation of feature space; i.e., for all $\delta \in I\!R^n$, $\alpha + \delta \succcurlyeq^{mm}_{\Lambda} \beta + \delta$ iff $(\alpha + \delta) \cdot \omega^{\star}_{\Lambda} \geq (\beta + \delta) \cdot \omega^{\star}_{\Lambda}$ if and only if $\alpha \succcurlyeq^{mm}_{\Lambda} \beta$. Therefore, in this section we only consider the effect of rescaling of feature spaces.

Let a *features rescaling* $\tau \in I\!R^n_+$ be a vector of strictly positive reals, with the jth component $\tau(j)$ being the scale factor for the jth feature. The effect of

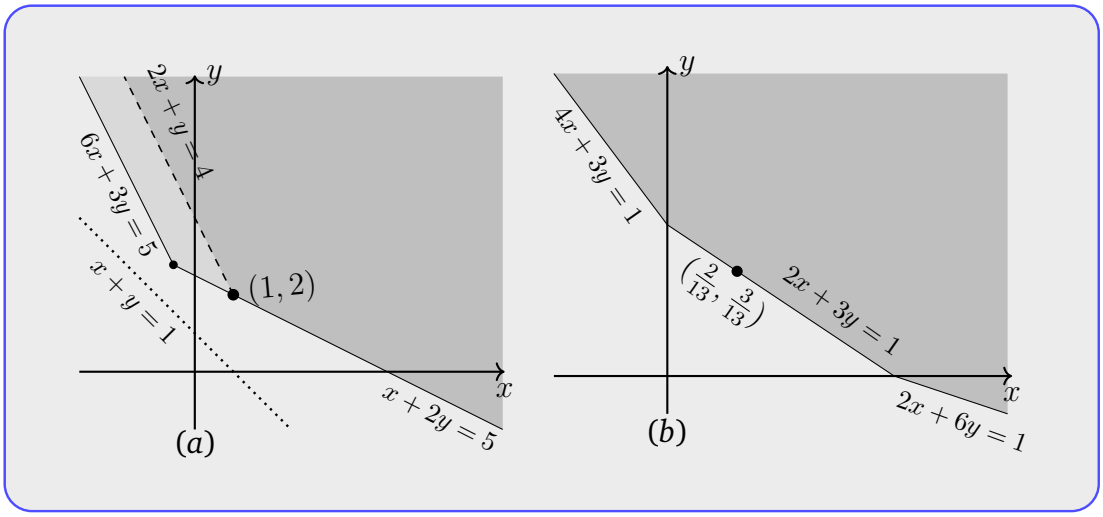

Figure 5.2: Some rescalings of inputs and features on $\Lambda = \{(2,1),(1,2),(1,1)\}$ are illustrated. **(a)** $\Lambda^{\geq}_{\mathbf{t}}$ is the union of the two shaded regions when inputs are rescaled by $\mathbf{t} = (3/5, 1/5, 1)$. The darkly shaded region indicates $\Lambda^{\geq}_{\mathbf{t}'}$ where $\mathbf{t}' = (1/4, 1/5, 1/3)$. We have $\omega^{\star}_{\Lambda_{\mathbf{t}}} = \omega^{\star}_{\Lambda_{\mathbf{t}'}} = (1,2)$. The darkly shaded region is also equal to $\Lambda^* + \{(1,2)\}$. **(b)** The shaded region indicates $(\Lambda \odot \tau)^{\geq}$ when features are rescaled by $\tau = (2,3)$. Here, $\omega^{\star}_{\Lambda\odot\tau} = (2/13, 3/13)$.

the rescaling on a vector $\lambda \in \mathbb{R}^n$ is given by pointwise multiplication, $\lambda \odot \tau$, defined by, for all $j = 1,\ldots,n$, $(\lambda \odot \tau)(j) = \lambda(j)\tau(j)$. Operation $\odot$ is commutative, associative and distributes over addition of vectors. An important property is that for any $u, v, w \in \mathbb{R}^n$ $(u \odot v) \cdot w = v \cdot (u \odot w)$, since they are both equal to $\sum_{j=1}^{n} u(j)v(j)w(j)$. For $\tau \in \mathbb{R}^n_+$, we define τ^{-1} to be the element of $\mathbb{R}^n_+$ given by $\tau^{-1}(j) = 1/\tau(j)$ for all $j \in \{1,\ldots,n\}$. The rescaling vector changes the preference inputs Λ, turning it into $\Lambda \odot \tau = \{\lambda \odot \tau : \lambda \in \Lambda\}$. Let $\omega^{\star}_{\Lambda\odot\tau}$ be the element with minimum norm in $(\Lambda \odot \tau)^{\geq}$, where $(\Lambda \odot \tau)^{\geq} = \{w \in \mathbb{R}^n : \forall \lambda \in \Lambda, w \cdot (\lambda \odot \tau) \geq 1\}$.

Example 15 » The Effect of Scaling of Features

Consider $\tau = (2,3)$ rescaling features of Λ in Example 13. Then, $\Lambda \odot \tau$ will be $\{(4,3),(2,6),(2,3)\}$. The shaded region in Figure 5.2(b) shows $(\Lambda \odot \tau)^{\geq}$, and it can be seen that $\omega^{\star}_{\Lambda\odot\tau} = (2/13, 3/13)$. Then, $(2,-1) \odot \tau \not\succcurlyeq^{mm}_{\Lambda\odot\tau} (0,0) \odot \tau$ because $(2,-1) \odot (2,3) \cdot (2/13, 3/13) = -1/13 < 0$, whereas $(2,-1) \succcurlyeq^{mm}_{\Lambda} (0,0)$ because $(2,-1) \cdot (1/2, 1/2) = 1/2 > 0$.

Like rescaling of inputs, we see that α might be preferred to β under one rescaling of features, but not under another. However, the choice of how the features are scaled relative to each other can involve somewhat arbitrary choices. It is therefore natural to consider a more cautious preference relation, *features-scaling-invariant* preference relation, given by α being preferred to β for all rescalings $\tau \in \mathbb{R}^n_+$.

Definition 33: Features-scaling-invariant Preference Relation, $\succcurlyeq^{\mathrm{F}}_{\Lambda}$

We define relation $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ by, for $\alpha, \beta \in \mathbb{R}^n$, $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta$ if and only if α is max-margin-preferred to β over all rescalings of features, i.e., if for all $\tau \in \mathbb{R}^n_+$, we have $\alpha \odot \tau \succcurlyeq^{mm}_{\Lambda \odot \tau} \beta \odot \tau$.

Now, define $\mathrm{SF}(\Lambda)$ to be $\{\omega^{\star}_{\Lambda \odot \tau} \odot \tau \,:\, \tau \in \mathbb{R}^n_+\}$; then we have:

Proposition 18

For any $\alpha, \beta \in \mathbb{R}^n$, we have $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta \iff$ for all $w \in \mathrm{SF}(\Lambda)$, $\alpha \cdot w \geq \beta \cdot w$.

Proof: We have that $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta \iff$ for all $\tau \in \mathbb{R}^n_+$, $\alpha \odot \tau \succcurlyeq^{mm}_{\Lambda \odot \tau} \beta \odot \tau$. Now, $\alpha \odot \tau \succcurlyeq^{mm}_{\Lambda \odot \tau} \beta \odot \tau$ if and only if $(\alpha \odot \tau) \cdot \omega^{\star}_{\Lambda \odot \tau} \geq (\beta \odot \tau) \cdot \omega^{\star}_{\Lambda \odot \tau}$, i.e., $\alpha \cdot (\omega^{\star}_{\Lambda \odot \tau} \odot \tau) \geq \beta \cdot (\omega^{\star}_{\Lambda \odot \tau} \odot \tau)$. Thus, $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta \iff$ for all $\tau \in \mathbb{R}^n_+$, $\alpha \cdot (\omega^{\star}_{\Lambda \odot \tau} \odot \tau) \geq \beta \cdot (\omega^{\star}_{\Lambda \odot \tau} \odot \tau)$, which is if and only if for all $w \in \mathrm{SF}(\Lambda)$, $\alpha \cdot w \geq \beta \cdot w$. ■

5.4.1 Rescale Optimality

We define an important notion, rescale optimality, for understanding the set $\mathrm{SF}(\Lambda)$, and hence the preference relation $\succcurlyeq^{\mathrm{F}}_{\Lambda}$.

Definition 34: Rescale-optimal

For $G \subseteq \mathbb{R}^n$, and $u \in G$, let us say that u *is rescale-optimal in* G if there exists (strictly positive) $\tau \in \mathbb{R}^n_+$ with $\|\tau \odot w\| \geq \|\tau \odot u\|$ for all $w \in G$.

It can be seen intuitively that elements of the (open) line segment between $(1,0)$ and $(0,1)$ in Figure 5.1(b) is the set of rescale-optimal elements in $\Lambda^{\geq}$; if $\tau(1) > \tau(2)$ (i.e., with the ratio $\tau(1)/\tau(2)$ being increased) then $\omega^{\star}_{\Lambda\odot\tau} \odot \tau$ moves from $(1/2, 1/2)$ towards $(1,0)$. Similarly, increasing the ratio $\tau(2)/\tau(1)$ from 1 moves $\omega^{\star}_{\Lambda\odot\tau} \odot \tau$ from $(1/2, 1/2)$ towards $(0,1)$.

We will show in Proposition 20 below that $\mathrm{SF}(\Lambda)$ is equal to the set of rescale-optimal elements in $\Lambda^{\geq}$. For instance, $\omega^{\star}_{\Lambda\odot\tau} \odot \tau$ in Example 15 (where $\tau(2)/\tau(1) = 3/2$), which is in $\mathrm{SF}(\Lambda)$ by the definition, is equal to $(2/13, 3/13) \odot (2,3) = (4/13, 9/13)$, that is between $(1/2, 1/2)$ and $(0,1)$. If $\mathrm{SF}(\Lambda)$ equals the line segment between $(1,0)$ and $(0,1)$, it can be seen that $(\mathrm{SF}(\Lambda))^*$ is the first quadrant in Figure 5.1(a). This implies that in Figure 5.1(a), $\gamma \succcurlyeq^{\mathrm{F}}_{\Lambda} \mathbf{0}$ if and only if γ is in the first quadrant. The following lemma is used to prove the equivalence in Proposition 20.

Lemma 19

Consider any $v \in \mathbb{R}^n$ and any $\tau \in \mathbb{R}^n_+$. Then, $v \in \Lambda^{\geq}$ if and only if $v \odot \tau^{-1} \in (\Lambda \odot \tau)^{\geq}$. Also, $w = v$ minimises $\|w \odot \tau^{-1}\|$ over $w \in \Lambda^{\geq}$ if and only if $v = \tau \odot \omega^{\star}_{\Lambda\odot\tau}$.

For example, we have seen in Example 15 that for $\tau = (2,3)$, $\omega^{\star}_{\Lambda\odot\tau} = (2/13, 3/13)$. This lemma implies that for $v = \tau \odot \omega^{\star}_{\Lambda\odot\tau} = (4/13, 9/13)$, there is no $u \in \Lambda^{\geq}$ such that $\|u \odot \tau^{-1}\| < \|v \odot \tau^{-1}\|$, i.e., v will be rescale-optimal in $\Lambda^{\geq}$.

Proof: The first part follows easily from the definitions. Regarding the second part, by definition of $\omega^{\star}_{\Lambda\odot\tau}$, we have that $v = \tau \odot \omega^{\star}_{\Lambda\odot\tau}$ if and only if $v \odot \tau^{-1}$ has minimum norm in $(\Lambda \odot \tau)^{\geq}$, which is if and only if

$w' = v \odot \tau^{-1}$ minimises $\|w'\|$ over $w' \in \{w' : \forall \lambda \in \Lambda, w' \cdot \lambda \odot \tau \geq 1\}$. By substituting w' with $w \odot \tau^{-1}$ this holds if and only if $v = w$ minimises $\|w \odot \tau^{-1}\|$ over $w \in \Lambda^{\geq}$. ■

Proposition 20

Consider any finite $\Lambda \subseteq \mathbb{R}^n$. $\mathrm{SF}(\Lambda)$ is equal to the set of all rescale-optimal elements of $\Lambda^{\geq}$. Thus, for $\alpha, \beta \in \mathbb{R}^n$, $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta$ if and only if $w \cdot (\alpha - \beta) \geq 0$ for every rescale-optimal element w in $\Lambda^{\geq}$.

Proof: Consider any $u \in \mathbb{R}^n$. Then, u is rescale-optimal in $\Lambda^{\geq}$ if and only if there exists $\tau \in \mathbb{R}^n_+$ such that $u = w$ minimises $\|w \odot \tau\|$ over $w \in \Lambda^{\geq}$, which, by Lemma 19, is if and only if there exists $\tau \in \mathbb{R}^n_+$ such that $u = \tau^{-1} \odot \omega^{\star}_{\Lambda \odot \tau^{-1}}$, which is, from the definition of $\mathrm{SF}(\Lambda)$, if and only if $u \in \mathrm{SF}(\Lambda)$. The last part follows from the first part and Proposition 18. ■

Proposition 20 implies, in particular, that $\omega^{\star}_{\Lambda}$ is rescale-optimal in $\Lambda^{\geq}$.

5.4.2 Pointwise Undominated

For $u \in G$, if there exists $v \in G$ such that for all j, $v(j)$ is between $u(j)$ and 0 then it is easy to see that u cannot be rescale-optimal element in G. This is the idea behind being pointwise undominated, which is reminiscent of being Pareto undominated.

Definition 35: Pointwise Dominance

For $u, v \in \mathbb{R}^n$, v pointwise dominates u if $u \neq v$ and for all $j \in \{1, \ldots, n\}$, either

(i) $0 \leq v(j) \leq u(j)$, or

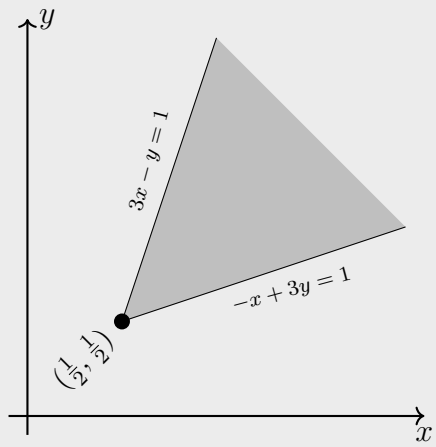

Figure 5.3: The visual representation of Example 16 when $\Lambda = \{(-1,3),(3,-1)\}$. $\Lambda^{\geq}$ is the shaded region with a single extremal point at $(1/2, 1/2)$. In this case, $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ is equal to $\succcurlyeq_{\Lambda}^{mm}$.

(ii) $0 \geq v(j) \geq u(j)$.

For $u \in G \subseteq \mathbb{R}^n$, we say that u is *pointwise undominated* in G if there exists no $v \in G$ that pointwise dominates u.

In Figure 5.1(b), all elements on the part of the *closed* line segment $x + y = 1$ within the first quadrant (i.e., including points on the axes) are pointwise undominated in $\Lambda^{\geq}$. The definition easily implies that being rescale-optimal implies being pointwise undominated (but not the converse).

Proposition 21

Let $G \subseteq \mathbb{R}^n$. If u is rescale-optimal in G then u is pointwise undominated in G. Thus, if u is pointwise dominated in G (i.e., there exists an element in G that pointwise dominates u) then u is not rescale-optimal in G.

Proof: Suppose that u is not pointwise undominated in G, so that there exists $v \in G$ that pointwise dominates u. Then, for every $j \in \{1,\ldots,n\}$,

$|v(j)| \leq |u(j)|$, and for some $k \in \{1, \ldots, n\}$, $|v(k)| < |u(k)|$, which implies that for every $\tau \in \mathbb{R}^n_+$, $\|v \odot \tau\| < \|u \odot \tau\|$, and hence, u is not rescale-optimal in G. ■

Proposition 21 states that being pointwise undominated is a necessary condition for being rescale-optimal. However, by having a look at our running example we will see that this not a sufficient condition. The intersection points of $x + y = 1$ with the axes (i.e., $(1, 0)$ and $(0, 1)$) are pointwise undominated but not rescale-optimal in $\Lambda^{\geq}$. To see this, suppose that for example $(1, 0)$ were rescale-optimal in $\Lambda^{\geq}$; i.e., there exists $\tau \in \mathbb{R}^2_+$ such that for all $v \in \Lambda^{\geq} \setminus \{(1, 0)\}$, $\|(1, 0) \odot \tau\| < \|v \odot \tau\|$. In particular, there exists $\tau \in \mathbb{R}^2_+$ such that for all $\epsilon \in (0, 1]$, $\|(1, 0) \odot \tau\| < \|(1 - \epsilon, \epsilon) \odot \tau\|$, i.e., $\tau^2(x) < (1 - \epsilon)^2\tau^2(x) + \epsilon^2\tau^2(y)$. Letting $r = \tau(x)/\tau(y)$, we obtain, there exists $r \in \mathbb{R}_+$ such that for all $\epsilon \in (0, 1]$, $r^2 < (1 - \epsilon)^2 r^2 + \epsilon^2$, and thus, $r^2 < \epsilon^2/(1 - (1 - \epsilon)^2) = \epsilon/(2 - \epsilon)$. Now, for any $r \in \mathbb{R}_+$ there exists sufficiently small $\epsilon > 0$ such that $\epsilon/(2 - \epsilon) < r^2$, proving that $(1, 0)$ is not rescale-optimal in $\Lambda^{\geq}$ by contradiction. We can use a similar argument to show that $(0, 1)$ is not rescale-optimal in $\Lambda^{\geq}$ as well. We will investigate this further in Section 5.4.4, leading to a computational procedure for rescale-optimality. First, in Section 5.4.3, we characterise the situations when rescaling of features makes no difference, in which case $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ is the same as $\succcurlyeq^{mm}_{\Lambda}$.

5.4.3 Determining Invariance to Rescaling of Features

Example 16 below illustrates that allowing rescaling of features can sometimes make no difference in maximum margin relation.

Example 16 » Invariance to Rescaling of Features

Consider Λ *be* $\{(-1, 3), (3, -1)\}$ *so that* $\Lambda^{\geq} = \{(x, y) : -x + 3y \geq 1, 3x - y \geq 1\}$ *which is the shaded region in Figure 5.3. Here,* $\Lambda^{\geq}$ *has a single extremal point at* $(1/2, 1/2)$*. Since* $(1/2, 1/2)$ *is the element with minimal norm in* $\Lambda^{\geq}$*,* $\omega^{\star}_{\Lambda} = (1/2, 1/2)$*, and so* $(1/2, 1/2)$ *is rescale-optimal in* $\Lambda^{\geq}$*. Also, all other points in* $\Lambda^{\geq}$ *are pointwise dominated by* $(1/2, 1/2)$*; thus,*

by Proposition 21, they are not rescale-optimal. Consequently, the only element of $\Lambda^{\geq}$ that is rescale-optimal is $(1/2, 1/2)$.

Note that if there exists a unique rescale-optimal element in $\Lambda^{\geq}$, then this element must be $\omega^{\star}_{\Lambda}$, since the latter is rescale-optimal by Proposition 20. This immediately implies that $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ is then equal to $\succcurlyeq^{mm}_{\Lambda}$.

Theorem 25 below states that u is the only rescale-optimal element in convex closed G if and only if u pointwise dominates every other element of G. The proof uses a triple of lemmas.

Lemma 22

Let G be a convex and (topologically) closed subset of $\mathbb{R}^n$. For each vector $\tau \in \mathbb{R}^n_+$, there exists a unique $w \in G$ with minimum value of $\|w \odot \tau\|$.

Proof: It is a standard result (for a proof see e.g., Proposition 4 of [WM16b]) that there is a unique element in a convex closed set with minimum norm. Consider any $\tau \in \mathbb{R}^n_+$. Now, $G \odot \tau = \{w \odot \tau : w \in G\}$ is convex and closed so there exists a unique element $w \odot \tau \in G \odot \tau$ with minimum value of $\|w \odot \tau\|$, so there is a unique $w \in G$ with minimum value of $\|w \odot \tau\|$. ■

Lemma 23

Let $u, v \in \mathbb{R}^n$. There exists $k \in \{1, \ldots, n\}$ such that $|u(k)| < |v(k)|$ if and only if there exists $\tau \in \mathbb{R}^n_+$ such that $\|u \odot \tau\| < \|v \odot \tau\|$. Thus, for all $j \in \{1, \ldots, n\}$, $|u(j)| \geq |v(j)|$ if and only if for all $\tau \in \mathbb{R}^n_+$, $\|u \odot \tau\| \geq \|v \odot \tau\|$.

Lemma 24

Let G be a convex subset of $\mathbb{R}^n$, and let j be any element of $\{1, \ldots, n\}$. Then either

(i) there exists $w \in G$ such that $w(j) = 0$; or

(ii) for all $w \in G$, $w(j) > 0$; or

(iii) for all $w \in G$, $w(j) < 0$.

Theorem 25: Uniquely Rescale-optimality

Let G be a convex and closed subset of $\mathbb{R}^n$, and let u be an element of G. Then the following conditions are equivalent.

(i) u is uniquely rescale-optimal in G, i.e., u is the unique element of G that is rescale-optimal;

(ii) for all $v \in G$, for all $j \in \{1, \ldots, n\}$, $|v(j)| \geq |u(j)|$;

(iii) u pointwise dominates every element in $G \setminus \{u\}$.

Consider Λ as it is in Example 16. Then, the three conditions hold for $u = (1/2, 1/2)$. The equivalence between (i) and (ii) is proved using Lemmas 22 and 23, and the equivalence between (ii) and (iii) follows using Lemma 24.

Proof: First suppose (i), that u is uniquely rescale-optimal in G, so that, for all $\tau \in \mathbb{R}^n_+$, and for all $v \in G \setminus \{u\}$, $\|u \odot \tau\| \leq \|v \odot \tau\|$; thus, by Lemma 23, for all $j \in \{1, \ldots, n\}$, $|u(j)| \leq |v(j)|$, showing that (ii) holds. The converse follows easily: if (ii) holds then for all $\tau \in \mathbb{R}^n_+$, for all $v \in G$, $\|u \odot \tau\| \leq \|v \odot \tau\|$, which by Lemma 22, leads to for all $\tau \in \mathbb{R}^n_+$, for all $v \in G \setminus \{u\}$, $\|u \odot \tau\| < \|v \odot \tau\|$, and thus proving (i).

We will next show that (ii) implies (iii). Assume that (ii) holds, and consider any $j \in \{1, \ldots, n\}$. We must show that for each $v \in G \setminus \{u\}$, either

$0 \leq u(j) \leq v(j)$ or $0 \geq u(j) \geq v(j)$ hold. This holds trivially if $u(j) = 0$, so suppose $u(j) \neq 0$. Then, (ii) implies that, for all $v \in G$, $v(j) \neq 0$, and thus, by Lemma 24, either, for all $v \in G$, $v(j) > 0$, or for all $v \in G$, $v(j) < 0$. (ii) then implies that for all $v \in G$, either $0 \leq u(j) \leq v(j)$ or $0 \geq u(j) \geq v(j)$, proving (iii). It immediately follows that (iii) implies (ii), completing the proof of equivalence of (i), (ii) and (iii). ■

Corollary 26 below leads immediately to an algorithm for determining if $\Lambda^{\geq}$ has a uniquely rescale-optimal element, and finding it, if it exists. The algorithm involves at most $n+1$ runs of a linear programming solver, and thus determining and finding a uniquely rescale-optimal element u can be performed in polynomial time. If it succeeds in finding such a u then the induced preferences can be efficiently tested using: $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta$ if and only if $u \cdot (\alpha - \beta) \geq 0$.

Corollary 26: Computation of Uniquely Rescale-optimal Element

Consider any finite $\Lambda \subset I\!R^n$, Choose an arbitrary element $y \in \Lambda^{\geq}$. Using y we will generate an element $y^* \in I\!R^n$. For each $j \in \{1, \ldots, n\}$: If $y(j) = 0$ then let $y^*(j) = 0$. If $y(j) > 0$ then let $y^*(j) = \inf\{w(j) : w \in \Lambda^{\geq}, w(j) \geq 0\}$. If $y(j) < 0$ then let $y^*(j) = \sup\{w(j) : w \in \Lambda^{\geq}, w(j) \leq 0\}$. If $y^* \in \Lambda^{\geq}$ then y^* is uniquely rescale-optimal in $\Lambda^{\geq}$. Also, there exists a uniquely rescale-optimal element in $\Lambda^{\geq}$ if and only if $y^* \in \Lambda^{\geq}$.

Consider Λ as in Example 16. Choose $y = (1, 1)$ which is in $\Lambda^{\geq}$. Since $y(1) = 1 > 0$, $y^*(1) = \inf\{w(1) : w \in \Lambda^{\geq}, w(1) \geq 0\} = \frac{1}{2}$. Similarly, $y(2) = 1$ implies that $y^*(2) = \frac{1}{2}$. Because $(\frac{1}{2}, \frac{1}{2}) \in \Lambda^{\geq}$, it is uniquely rescale-optimal in $\Lambda^{\geq}$. If you consider Λ as in Example 13, then $y = (1, 1)$ will be updated to $y^* = (0, 0)$, and because $(0, 0) \notin \Lambda^{\geq}$, there does not exist a uniquely rescale-optimal element in $\Lambda^{\geq}$.

Proof: Consider any $j \in \{1, \ldots, n\}$. Lemma 24 implies that if $y(j) > 0$ then for all $w \in \Lambda^{\geq}$, $0 \leq y^*(j) \leq w(j)$; and if $y(j) < 0$ then for all $w \in \Lambda^{\geq}$, $0 \geq y^*(j) \geq w(j)$, so for all $w \in \Lambda^{\geq}$, either $0 \leq y^*(j) \leq w(j)$ or $0 \geq y^*(j) \geq w(j)$. Theorem 25 implies that if $y^* \in \Lambda^{\geq}$ then y^* is uniquely

rescale-optimal in $\Lambda^{\geq}$.

Conversely, suppose that there exists a uniquely rescale-optimal element u in $\Lambda^{\geq}$; we will prove that $y^* \in \Lambda^{\geq}$. Consider arbitrary $j \in \{1, \ldots, n\}$. The fact that $\Lambda^{\geq}$ is a polyhedron implies that there exists some $w \in \Lambda^{\geq}$ with $w(j) = y^*(j)$. If $y^*(j) > 0$ then we know by Lemma 24 that $u(j) \geq y^*(j)$. But Theorem 25 implies that $u(j) \leq w(j) = y^*(j)$, and thus $y^*(j) = u(j)$. Similarly, if $y^*(j) < 0$ then $y^*(j) \geq u(j) \geq w(j) = y^*(j)$ and so, $y^*(j) = u(j)$. If $y^*(j) = 0$ then $w(j) = 0$, so $u(j) = 0$, also using Theorem 25. We have shown that $y^* = u$, so $y^* \in \Lambda^{\geq}$. ■

5.4.4 Characterising Rescale-Optimality

As we illustrated, being pointwise undominated is not a sufficient condition for being rescale-optimal. In this section we define a stronger version of pointwise undominated called *zm-pointwise undominated*, where 'zm' stands for *zeros-modified* (the essential difference being in the treatment of j such that $u(j) = 0$). We show that this is still a necessary condition, and is in fact equivalent to being rescale-optimal (for polyhedra). According to the following definition, while the points $(1, 0)$ and $(0, 1)$ in Figure 5.1(b) were pointwise undominated, they are not zm-pointwise undominated.

Definition 36: zm-pointwise Dominance

For $u, v \in I\!R^n$, let $N_u = \{j \in \{1, \ldots, n\} : u(j) \neq 0\}$. v zm-pointwise dominates u if there exists $k \in N_u$ such that $u(k) \neq v(k)$ and for all $j \in N_u$, either (i) $0 \leq v(j) \leq u(j)$, or (ii) $0 \geq v(j) \geq u(j)$.

For $u \in G \subseteq I\!R^n$, we say that u is *zm-pointwise undominated* in G if there exists no $v \in G$ that zm-pointwise dominates u.

The definitions easily imply the following lemma, which relates pointwise dominance and zm-pointwise dominance.

Lemma 27

Consider any $u, v \in \mathbb{R}^n$. If v pointwise dominates u then v zm-pointwise dominates u.

Now suppose that $u \in G \subseteq \mathbb{R}^n$. If u is zm-pointwise undominated in G then u is pointwise undominated in G. In addition, the converse holds if none of the components of u is zero.

Proof: Suppose that v pointwise dominates u. Then, $u \neq v$ and for all $j \in \{1, \ldots, n\}$, either (i) $0 \leq v(j) \leq u(j)$, or (ii) $0 \geq v(j) \geq u(j)$. So, for some $k \in \{1, \ldots, n\}$, $u(k) \neq v(k)$, which implies that $u(k) \neq 0$. This implies that v zm-pointwise dominates u. The other two parts follow immediately from the definitions. ■

Lemma 28 below characterises the zm-pointwise undominated elements in a convex set.

Lemma 28

Consider any convex set $G \in \mathbb{R}^n$. Then, u is zm-pointwise undominated in convex G if and only if for all $v \in G$, either

(i) $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$; or

(ii) there exists $k \in \{1, \ldots, n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$.

Proposition 30 below shows that being zm-pointwise undominated is a necessary condition for being rescale-optimal. The proof uses the following lemma.

Lemma 29

Let $u, v \in \mathbb{R}^n$, with $u \neq v$, and let $\tau \in \mathbb{R}^n_+$. For $\delta \in (0, 1]$ let $v_\delta = \delta v + (1-\delta)u$. Then the following hold:

(i) For any $\delta \in (0, 1]$, $\|v_\delta \odot \tau\|^2 - \|u \odot \tau\|^2 = \delta^2 \|(v - u) \odot \tau\|^2 + 2\delta(\tau \odot \tau \odot u) \cdot (v - u)$.

(ii) $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$ if and only if for all $\delta \in (0, 1]$, $\|v_\delta \odot \tau\| > \|u \odot \tau\|$.

(iii) There exists $\tau \in \mathbb{R}^n_+$ such that $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$ if and only if either (a) $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$; or (b) there exists $k \in \{1, \ldots, n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$.

Proposition 30

Let u be an element of convex $G \subseteq \mathbb{R}^n$. Then:

(i) u is rescale-optimal in G if and only if there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G$, $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$.

(ii) u is zm-pointwise undominated in G if and only if for all $v \in G$, there exists $\tau \in \mathbb{R}^n_+$ such that $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$.

(iii) If u is rescale-optimal in G then u is zm-pointwise undominated in G.

Proof: (i): Using Lemma 22, u is rescale-optimal in G if and only if there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G \setminus \{u\}$, $\|v \odot \tau\| > \|u \odot \tau\|$, which, since G is convex, is if and only if, there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G \setminus \{u\}$ and for all $\delta \in (0, 1]$, $\|v_\delta \odot \tau\| > \|u \odot \tau\|$, where $v_\delta = \delta v + (1-\delta)u$. By Lemma 29(ii), this is if and only if there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G \setminus \{u\}$, $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$, which holds if and only if there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G$, $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$.

(ii) By Lemma 28 and Lemma 29(iii), u is zm-pointwise undominated in G if and only if for all $v \in G \setminus \{u\}$, there exists $\tau \in \mathbb{R}^n_+$ such that $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$, from which (ii) follows.

(iii) follows immediately from (i) and (ii). ■

We say that $u, v \in \mathbb{R}^n$ *agree on signs* if, for each component j, $u(j)$ and $v(j)$ have equal sign.

Definition 37: Agreeing on Signs

For $u, v \in \mathbb{R}^n$, u *and* v *agree on signs* if for all $j = 1, \ldots, n$,

(i) $u(j) = 0 \iff v(j) = 0$;

(ii) $u(j) > 0 \iff v(j) > 0$; and thus also:

(iii) $u(j) < 0 \iff v(j) < 0$.

For example, $(1, 0)$ and $(1, 1)$ do not agree on signs but for $\varepsilon > 0$, $(1, \varepsilon)$ and $(1, 1)$ agree on signs. The following is the key theorem of this section to characterise rescale-optimality by making use of Proposition 30(i).

Theorem 31: Rescale-optimality Characterisation

Consider any u in convex $G \subseteq \mathbb{R}^n$. If $u = \mathbf{0}$ then it is the unique rescale-optimal element of G. Otherwise, u is rescale-optimal in G if and only if there exists $\mu \in \mathbb{R}^n$ agreeing on signs with u such that $\mu \cdot u = 1$ and for all $w \in G$, $\mu \cdot w \geq 1$.

To illustrate this result, consider u be any element of the part of the open line segment $x + y = 1$ within the first quadrant in Figure 5.1(b); i.e., $u = (\delta, 1 - \delta)$ for some $\delta \in (0, 1)$. Then for $\mu = (1, 1)$, μ and u agree on signs and $u \cdot \mu = 1$, and for all $w \in \Lambda^{\geq}$, $\mu \cdot w \geq 1$. Therefore, this theorem implies that u is rescale-optimal in $\Lambda^{\geq}$.

Proof: It is clear that if $u = \mathbf{0}$ then for all $\tau \in \mathbb{R}^n_+$ and for all $w \in G \setminus \{u\}$, $\|u \odot \tau\| = 0 < \|w \odot \tau\|$, which means that u is the unique rescale-optimal

element of G. Now, suppose $u \neq \mathbf{0}$.

$\Rightarrow$: First, let us assume that u is rescale-optimal in G. Then, by Proposition 30(i), there exists $\tau \in \mathbb{R}^n_+$ such that for all $w \in G$, $(\tau \odot \tau \odot u) \cdot (w - u) \geq 0$. Let $\mu' = \tau \odot \tau \odot u$. Because, $u \neq \mathbf{0}$, $\mu' \cdot u = \|\tau \odot u\|^2 > 0$. Then, we let $\mu = \frac{\mu'}{\mu' \cdot u}$ which leads to $\mu \cdot u = 1$. In addition, μ and μ' agree on signs because $\mu' \cdot u > 0$, and μ' and u agree on signs, by the definition of μ', and hence μ and u agree on signs. Consider any $w \in G$. We know that $(\tau \odot \tau \odot u) \cdot (w - u) \geq 0$, i.e., $\mu' \cdot (w - u) \geq 0$, which implies that $\mu \cdot (w - u) \geq 0$, and therefore, $\mu \cdot w \geq 1 = \mu \cdot u$.

$\Leftarrow$: For the converse, we assume that there exists $\mu \in \mathbb{R}^n$ agreeing on signs with u such that $\mu \cdot u = 1$ and for all $w \in G$, $\mu \cdot w \geq 1$, and thus, $\mu \cdot (w - u) \geq 0$. Define $\tau \in \mathbb{R}^n$ by $\tau(j) = 1$ if $u(j) = 0$, and $\tau(j) = \sqrt{\frac{\mu(j)}{u(j)}}$ whenever $u(j) \neq 0$, which is well-defined, because $\mu(j)/u(j) > 0$ whenever $u(j) \neq 0$, using the fact that μ and u agree on signs. That fact also means that $u(j) = 0$ implies that $\mu(j) = 0$. We then have that for all $j \in \{1, \ldots, n\}$, $\tau(j)^2 u(j) = \mu(j)$, which implies that $\tau \odot \tau \odot u = \mu$. Therefore, for all $w \in G$, $(\tau \odot \tau \odot u) \cdot (w - u) \geq 0$, which implies that u is rescale-optimal in G, by Proposition 30(i). ■

5.4.5 Equivalence of Rescale-Optimal With Zm-Pointwise Undominated

It turns out that being zm-pointwise undominated is equivalent to being rescale-optimal, for a polyhedron; see Theorem 36 below. Regarding the definition of polyhedron in Chapter 2, we can write any polyhedron as $G_I = \{w \in \mathbb{R}^n : \forall i \in I, w \cdot \lambda_i \geq a_i\}$, for finite I, and with each $\lambda_i \in \mathbb{R}^n$ and $a_i \in \mathbb{R}$. We also consider $G_{J_u} = \{w \in \mathbb{R}^n : \forall i \in J_u, w \cdot \lambda_i \geq a_i\}$, where $J_u = \{i \in I : \lambda_i \cdot u = a_i\}$. Clearly, for all $u \in G_I$, $G_I \subseteq G_{J_u}$.

For example, consider $a_1 = a_2 = 1$, $u = (1, 0)$, $v = (1/2, 1/2)$ and $y = (1, 1)$, with the vectors λ_i for $i \in I = \{1, 2, 3\}$ being as in Example 13 and Figure 5.1. Then, $G_I = \Lambda^{\geq}$; $G_{J_u} = \{w \in \mathbb{R}^2 : w \cdot (1, 1) \geq 1, w \cdot (1, 2) \geq 1\}$; $G_{J_v} = \{w \in \mathbb{R}^2 : w \cdot (1, 1) \geq 1\}$; and $G_{J_y} = \mathbb{R}^2$ because $J_y = \emptyset$.

We next give some lemmas that we will use in the proof of Theorem 36.

Lemma 32

Consider any $u \in G_I$ and any $v \in G_{J_u}$. Then there exists $\delta' \in (0,1)$ such that for all δ with $0 < \delta \leq \delta'$, $\delta v + (1-\delta)u \in G_I$.

To illustrate, consider $u = (1,0)$ and $v = (-1,2)$ which is in G_{J_u}. We can see in Figure 5.1(b) that the line segment from u to $(0,1)$ is in G_I but beyond that from $(0,1)$ to v is not. That means that choosing $\delta' = 1/2$ works for this case (because $\frac{1}{2}v + (1-\frac{1}{2})u = (0,1)$); i.e., for all δ with $0 < \delta \leq 1/2$, $\delta v + (1-\delta)u \in G_I$.

Lemma 33

Consider non-zero $u \in G_I$ (as defined above). Then u is zm-pointwise undominated in G_I if and only if u is zm-pointwise undominated in G_{J_u}.

Lemma 34

$G_{J_u} + \{-u\}$ is equal to $\{\lambda_i : i \in J_u\}^*$.

$G_{J_u} + \{-u\}$ means translating G_{J_u} to move u to the origin. So, continuing the example, for $u = (1,0)$, $G_{J_u} + \{-u\} = \{w \in I\!R^2 : w \cdot (1,1) \geq 0, w \cdot (1,2) \geq 0\} = \{(1,1),(1,2)\}^* = \{\lambda_i : i \in J_u\}^*$.

Proof: $v \in G_{J_u} + \{-u\}$ if and only if for all $i \in J_u$, $(v+u) \cdot \lambda_i \geq a_i$, which is if and only if for all $i \in J_u$, $v \cdot \lambda_i \geq 0$ (since, by definition, $u \cdot \lambda_i = a_i$ for all $i \in J_u$), which is if and only if $v \in (\{\lambda_i : i \in J_u\})^*$. ■

Lemma 35

For $u, v \in \mathbb{R}^n$, If u and v agree on signs and $u \neq \mathbf{0}$ then $u \cdot v > 0$.

Theorem 36: Equivalence of zm-pointwise Undominated and Rescale-optimal Elements

Let u be an element of polyhedron $G \subseteq \mathbb{R}^n$. Then, u is rescale-optimal in G if and only if u is zm-pointwise undominated in G.

Proof:

G is a polyhedron, so, by definition, it can be written as $\{w \in \mathbb{R}^n : \forall i \in I, w \cdot \lambda_i \geq a_i\}$. Let $J_u = \{i \in I : \lambda_i \cdot u = a_i\}$, and let $G_{J_u} = \{w \in \mathbb{R}^n : \forall i \in J_u, w \cdot \lambda_i \geq a_i\}$. Proposition 30(iii) implies that if u is rescale-optimal in G then u is zm-pointwise undominated. We next prove the converse.

Assume that u is zm-pointwise undominated in G. Let $C = G_{J_u} + \{-u\}$. By Lemma 34, $C = \{\lambda_i : i \in J_u\}^*$, which is a polyhedral cone (i.e., a polyhedron that is cone), and thus, by the Minkowski-Weyl theorem (see e.g., Theorem 4.18 of [Gal08]), is a finitely generated convex cone, so we can write $C = co(W)$ for some finite set $W = \{w_1, \ldots, w_l\}$.

Let $C' = co(S)$ be the convex cone generated by $S = W \cup S_Z$ where $S_Z = \{e_j, -e_j : j \in Z\}$, and $e_j \in \mathbb{R}^n$ is the unit vector in the jth dimension, and $Z = \{j \in \{1, \ldots, n\} : u(j) = 0\}$. Also, let $T = E_+ \cup E_- \cup R$, where $E_+ = \{-e_j : u(j) > 0\}$, and $E_- = \{e_j : u(j) < 0\}$, and $R = \{-w_i : i \in M\}$, and where $M = \{i \in L : -w_i \notin C'\}$ and $L = \{1, \ldots, l\}$. Let H be the convex hull of T.

We will show that the assumption that u is zm-pointwise undominated implies that C' and H are disjoint. If there exists $h \in C' \cap H$ then h can be written as $w + v_0$ where $w \in C$ and $v_0 \in co(S_Z)$. Also, since $h \in H$, it can be written as $v_+ + v_- + y$, where $v_+ \in co(E_+)$, $v_- \in co(E_-)$ and $y \in co(R)$.

(More specifically, for some $q_1, q_2, q_3 \in [\,0, 1\,]$ with $q_1 + q_2 + q_3 = 1$ we have $v_+ = q_1 v'_+$ for some v'_+ in the convex hull of E_+, and $v_- = q_2 v'_-$ for some v'_- in the convex hull of E_-, and $y = q_3 z$ for some z in the convex hull of R.) Since $-y \in C$, $w - y \in C$. Let $v = w - y + u = -v_0 + v_+ + v_- + u$. Then $v \in G_{J_u}$, because $v - u = w - y \in C$.

For $j \in \{1, \ldots, n\}$, if $u(j) > 0$ then $v_0(j) = v_-(j) = 0$, so $v(j) = u(j) + v_+(j) \leq u(j)$. Similarly, if $u(j) < 0$ then $v(j) = u(j) + v_-(j) \geq u(j)$. Thus, if $u(j) > 0$ then $v(j) \leq u(j)$; and if $u(j) < 0$ then $v(j) \geq u(j)$. Since, u is zm-pointwise undominated in G, u is zm-pointwise undominated in G_{J_u}, by Lemma 33. Lemma 28 then implies that for all $j \in \{1, \ldots, n\}$, if $u(j) \neq 0$ then $v(j) = u(j)$, and thus $v_+(j) = v_-(j) = 0$, and so, $v_+ = v_- = \mathbf{0}$ (since also, if $u(j) = 0$ then $v_+(j) = v_-(j) = 0$, by definition of v_+ and v_-, and of E_+ and E_-). This implies that $w + v_0 = y$ and $y \in H$. Also, since $\mathbf{0}$ is neither in the convex hull of E_+ nor E_-, we have $q_1 = q_2 = 0$, and thus $q_3 = 1$, and so, y is in the convex hull of R. By definition of convex hull, we can write y as $\sum_{i \in M} t_i(-w_i)$, with each $t_i \geq 0$, and for some $k \in M$, $t_k > 0$. Then $-t_k w_k = w + \sum_{i \in M, i \neq k} t_i w_i + v_0$. The right-hand-side is in $co(S)$, which equals C', which implies that $-w_k \in C'$, which contradicts $k \in M$. Thus, C' and H are disjoint.

Both C' and H are convex and topologically closed, and H is compact. A strict separating hyperplane theorem (see e.g., Theorem 2.1.5 of [BGW09]) implies that there exists vector $\mu \in I\!R^n$ and $c \in I\!R$ such that for all $g \in C'$, $\mu \cdot g > c$ and for all $h \in H$, $\mu \cdot h < c$. Since $\mathbf{0} \in C'$, we have $\mu \cdot \mathbf{0} > c$, so $c < 0$.

Now, if g and $-g$ are both in C' then $\mu \cdot g = 0$. (Else $\mu \cdot g < 0$ or $\mu \cdot (-g) < 0$; without loss of generality assume $\mu \cdot g < 0$; then there exists $r > 0$ such that $\mu \cdot (rg) = r(\mu \cdot g) < c$, which contradicts $rg \in C'$.) This implies that if $u(j) = 0$ (so $j \in Z$ and $e_j, -e_j \in C'$) then $\mu \cdot e_j = 0$ and thus $\mu(j) = 0$. Also, if $i \in L - M$, then $w_i, -w_i \in C'$, so $\mu \cdot w_i = 0$. For any $i \in M$, we have that $-w_i \in H$, so $\mu \cdot (-w_i) < c < 0$, so $\mu \cdot w_i > 0$. Thus for any $w_i \in W$, $\mu \cdot w_i \geq 0$, and therefore for any $w \in C$, $\mu \cdot w \geq 0$, since w is a positive linear combination of the elements of W.

If $u(j) > 0$, then $-e_j \in H$, so $\mu \cdot e_j > -c > 0$, so $\mu(j) > 0$. Similarly, if $u(j) < 0$ then $\mu(j) < 0$. Thus, μ agrees on signs with u. This, by using Lemma 35, implies that $\mu \cdot u > 0$ (since $u \neq \mathbf{0}$), and we let $\mu' = \frac{\mu}{\mu \cdot u}$. So

$\mu' \cdot u = 1$, and μ' agrees on signs with μ and then u too.

For any $v \in G$, $v \in G_{J_u}$, and so $v - u$ is in C; we have shown that $\mu' \cdot (v - u) \geq 0$, so $\mu' \cdot v \geq \mu' \cdot u = 1$. Theorem 31 then implies that u is rescale-optimal in G. ■

5.4.6 Rescale-Optimality in Terms of Positive Linear Combinations

Here we extend the characterisation of rescale-optimality given in Theorem 31, leading to a computational method for testing rescale-optimality, and thus to a method for testing if $\alpha \succcurlyeq^{\mathrm{F}}_{\Lambda} \beta$, for $\alpha, \beta \in I\!R^n$. Theorem 31 implies that non-zero u is rescale-optimal in G if and only if there exists a vector μ that agrees on signs with u with $\mu \cdot w \geq \mu \cdot u$ for all $w \in G$. The main result of this section is Theorem 38, showing that μ is a positive linear combination of certain vectors when G is a polyhedron. Recall the definitions of $G_I = \{w \in I\!R^n : \forall i \in I, w \cdot \lambda_i \geq a_i\}$, $G_{J_u} = \{w \in I\!R^n : \forall i \in J_u, w \cdot \lambda_i \geq a_i\}$ and $J_u = \{i \in I : \lambda_i \cdot u = a_i\}$ from the previous section. The following lemma is used in the proof.

Lemma 37

Consider a polyhedron G_I and non-zero $u \in G_I$. Then u is rescale-optimal in G_I if and only if u is rescale-optimal in G_{J_u}.

This follows from Theorem 36 and Lemma 33, since G_I and G_{J_u} are polyhedra. However, we give a more direct proof in the appendix.

Theorem 38: Rescale-Optimality in Terms of Positive Linear Combinations

Let G be a polyhedron, which we write as $G_I = \{w \in I\!R^n : \forall i \in I, w \cdot \lambda_i \geq a_i\}$, for finite I, and with each $\lambda_i \in I\!R^n$ and $a_i \in I\!R$. Consider any non-zero vector u in G_I. Then, u is rescale-optimal in G_I if and only

if there exists $\mu \in I\!R^n$ that agrees on signs with u such that $\mu \cdot u = 1$ and $\mu \in co(\{\lambda_i : i \in J_u\})$.

Recall the example illustrating Theorem 31 where $u = (\delta, 1-\delta)$ for some (arbitrary) $\delta \in (0,1)$. We can see that the set $\{\lambda_i : i \in J_u\}$ equals $\{(1,1)\}$. So, choosing $\mu = (1,1)$ gives that u is rescale-optimal in $\Lambda^{\geq}$ because u and μ agree on signs, $u \cdot \mu = 1$, and $\mu \in co(\{(1,1)\})$.

Note that this theorem implies that if non-zero u is rescale-optimal in G_I then J_u is non-empty, since $\mathbf{0}$ is the only positive linear combination of the empty set, and $\mu \neq \mathbf{0}$.

Proof: First consider $\mu \in I\!R^n$ such that $\mu \cdot u = 1$. Then it can be seen that $\{w : w \cdot \mu \geq 1\} + \{-u\} = (\{\mu\})^*$. Thus, adding $-u$ to both sets gives $G_{J_u} \subseteq \{w : w \cdot \mu \geq 1\}$ if and only if $G_{J_u} + \{-u\} \subseteq (\{\mu\})^*$ which is if and only if $\{\lambda_i : i \in J_u\}^* \subseteq (\{\mu\})^*$, using Lemma 34, which, by Lemma 3, is if and only if, $\mu \in co(\{\lambda_i : i \in J_u\})$.

By Lemma 37, u is rescale-optimal in G_I if and only if u is rescale-optimal in G_{J_u}, which, by Theorem 31, is if and only if there exists $\mu \in I\!R^n$ agreeing on signs with u such that $\mu \cdot u = 1$ and $G_{J_u} \subseteq \{w : w \cdot \mu \geq 1\}$, i.e., $\mu \in co(\{\lambda_i : i \in J_u\})$, by the earlier argument. ■

We have the following corollary (using the same notation), which shows that testing if u is rescale-optimal in G_I can be performed in polynomial time: by first checking that $u \in G_I$ (i.e., for all $i \in I$, $u \cdot \lambda_i \geq a_i$), and then testing if a set of inequalities has a solution, using a linear programming solver.

Corollary 39: Computation of Rescale-optimality

Let u be a non-zero element of $I\!R^n$. Then, u is rescale-optimal in G_I if and only if $u \in G_I$ and there exists non-negative reals r_i for each $i \in J_u$, and vector $\tau \in I\!R^n$ with for all $j \in \{1, \ldots, n\}$, $\tau(j) \geq 1$, and $\tau(j)u(j) = \sum_{i \in J_u} r_i \lambda_i(j)$.

Proof: First suppose that u is rescale-optimal in G_I. Then $u \in G_I$, and, by Theorem 38, there exists $\mu \in I\!R^n$ that agrees on signs with u such that $\mu \cdot u = 1$ and there exist non-negative $r'_i \in I\!R$ such that $\mu = \sum_{i \in J_u} r'_i \lambda_i$. For all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$, define $t_j = \mu(j)/u(j)$, which is greater than zero, because μ and u agree on signs, and let t be the minimum of these values. Define τ by $\tau(j) = 1$ if $u(j) = 0$ and otherwise, define $\tau(j) = t_j/t$. Then for all $j \in \{1, \ldots, n\}$, $\tau(j) \geq 1$, and $\tau(j)u(j) = \mu(j)/t = \sum_{i \in J_u}(r'_i/t)\lambda_i(j)$.

Conversely, suppose that $u \in G_I$ and there exists non-negative reals r_i for each $i \in J_u$ and vector $\tau \in I\!R^n$ with for all $j \in \{1, \ldots, n\}$, $\tau(j) \geq 1$, and $\tau(j)u(j) = \sum_{i \in J_u} r_i \lambda_i(j)$. Define $\mu \in I\!R^n$ to be $\frac{\tau \odot u}{(\tau \odot u) \cdot u}$. Then $\mu \cdot u = 1$, and μ agrees on signs with u, and is a positive linear combination of $\{\lambda_i : i \in J_u\}$. Theorem 38 then can be applied to give the result. ■

Theorem 38 implies the following, which leads to a computational method for checking dominance with respect to $\succcurlyeq^{\mathrm{F}}_{\Lambda}$.

Theorem 40

Consider any finite $\Lambda \subseteq I\!R^n$ and any $u \in I\!R^n$. Define $\Theta_u = \{\lambda \in \Lambda : \lambda \cdot u = 1\}$. Then, u is in $\mathrm{SF}(\Lambda)$ if and only if $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$ such that μ agrees on signs with u, and $\mu \in co(\Theta_u)$. Also, u is in $\mathrm{SF}(\Lambda)$ if and only if $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$ and some subset Δ of Θ_u such that $|\Delta| \leq n + 1$, and $\mu \in co(\Delta)$, and μ agrees on signs with u.

Proof: Proposition 20 implies that $\mathrm{SF}(\Lambda)$ equals the set of all rescale-optimal elements of $\Lambda^{\geq}$. Hence, Theorem 38 implies that $u \in \mathrm{SF}(\Lambda)$ if and only if $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$ that agrees on signs with u such that $\mu \in co(\Theta_u)$ and $\mu \cdot u = 1$. First, we will show that the condition $\mu \cdot u = 1$ can be omitted.

Suppose first that $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$ that agrees on signs with u such that $\mu \in co(\Theta_u)$. Now, u is a non-zero vector, since $u \in \Lambda^{\geq}$.

Since μ agrees on signs with u, we have, by Lemma 35, $\mu \cdot u > 0$. Define $\mu' = \frac{\mu}{\mu \cdot u}$. Then $\mu' \in co(\Lambda)$, $\mu' \cdot u = 1$, and μ' and u agree on signs. We can then apply Theorem 38 to give $u \in \mathrm{SF}(\Lambda)$. The converse follows immediately from the same theorem.

The last part follows from Carathéodory's Theorem (see e.g., 3.1.2 in [NP06]) which states that for any $w \in I\!R^n$ and any $S \subseteq I\!R^n$, if $w \in co(S)$ then there exists $S' \subseteq S$ with $|S'| \leq n+1$ such that $w \in co(S')$. ■

Now, we have the following procedure to compute the $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ relation; Proposition 18 implies that for $\alpha, \beta \in I\!R^n$, $\alpha \not\succcurlyeq^{\mathrm{F}}_{\Lambda} \beta$ if and only if there exists $u \in \mathrm{SF}(\Lambda)$ such that $\alpha \cdot u < \beta \cdot u$, which, by Theorem 40, is if and only if there exists $u \in I\!R^n$ and $\mu \in I\!R^n$, such that (i) $u \cdot (\beta - \alpha) > 0$, (ii) for all $i \in I$, $u \cdot \lambda_i \geq 1$ (i.e., $u \in \Lambda^{\geq}$), (iii) for all $j = \{1, \ldots, n\}$, $u(j) = 0 \iff \mu(j) = 0$, and $u(j) > 0 \iff \mu(j) > 0$ (i.e., agreeing on signs), and (iv) there exists some subset Δ of Θ_u (as defined above) such that $|\Delta| \leq n+1$, and there exist non-negative reals r_i for each $i \in I$ such that $\mu = \sum_{i \in I} r_i \lambda_i$ where $r_i = 0$ if $\lambda_i \notin \Delta$ (i.e., $\mu \in co(\Delta)$). The (iv) part holds if and only if there exist non-negative reals r_i for each $i \in I$ such that for all $i \in I$, either $\lambda_i \in \Delta \Rightarrow u \cdot \lambda_i = 1$ or $\lambda_i \notin \Delta \Rightarrow r_i = 0$, $\sum_{i \in I}(r_i \neq 0) \leq n+1$, and $\mu = \sum_{i \in I} r_i \lambda_i$.

5.5 Simultaneous Rescaling of Features and Inputs

Having defined $\succcurlyeq^{\mathrm{I}}_{\Lambda}$ and $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ relations, it is also natural to consider both kinds of rescaling simultaneously: preference inputs and features. In this section, we define and characterise a preference relation based on both kinds of rescaling.

Definition 38: $\mathrm{SIF}(\Lambda)$ **and** $\succcurlyeq^{\mathrm{I,F}}_{\Lambda}$

We define the set $\mathrm{SIF}(\Lambda)$ by $w \in \mathrm{SIF}(\Lambda)$ if there exists $\mathbf{t} \in (0,1]^m$ such that $w \in \mathrm{SF}(\Lambda_{\mathbf{t}})$; i.e., $\mathrm{SIF}(\Lambda) = \{\omega^{\star}_{\Lambda_{\mathbf{t}} \odot \tau} \odot \tau : \mathbf{t} \in (0,1]^m, \tau \in I\!R^n_+\}$. We define relation $\succcurlyeq^{\mathrm{I,F}}_{\Lambda}$ by $\alpha \succcurlyeq^{\mathrm{I,F}}_{\Lambda} \beta \iff$ for all $w \in \mathrm{SIF}(\Lambda)$, $w \cdot \alpha \geq w \cdot \beta$.

This definition implies that $\alpha \succcurlyeq^{\mathrm{I,F}}_{\Lambda} \beta$ if and only if for all rescalings of the features and the preference inputs, α is max-margin preferred to β. We have the following characterisation, which leads to a computational method for checking if $\alpha \succcurlyeq^{\mathrm{I,F}}_{\Lambda} \beta$.

Theorem 41

$u \in \mathrm{SIF}(\Lambda)$ if and only if $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$ that agrees on signs with u such that $\mu \in co(\Lambda)$.

Consider Λ as in Example 13. For any choice of $\mu \in co(\Lambda)$, we have either $\mu = (0,0)$ or $\mu(1) > 0$ and $\mu(2) > 0$ (see Figure 5.1(a)). Because $(0,0) \notin \Lambda^{\geq}$, $u \in \mathrm{SIF}(\Lambda)$ if $u(1) > 0$ and $u(2) > 0$ (to agree on signs with μ); however, $u \neq (0,0)$ because $(0,0) \notin \Lambda^{\geq}$. As a result, $\mathrm{SIF}(\Lambda)$ is the part of $\Lambda^{\geq}$ (i.e., the darkly shaded region) that is *strictly* within the first quadrant in Figure 5.1(b). It can be seen that $(\mathrm{SIF}(\Lambda))^*$ is the first quadrant in Figure 5.1(a). This implies that for $\gamma \in I\!R^n$, $\gamma \succcurlyeq^{\mathrm{I,F}}_{\Lambda} \mathbf{0}$ if and only if γ is in the first quadrant in Figure 5.1(a).

Proof:

We first show that $\mathrm{SIF}(\Lambda) \subseteq \Lambda^{\geq}$. We have $\mathrm{SIF}(\Lambda) = \bigcup_{\mathbf{t}} \mathrm{SF}(\Lambda_{\mathbf{t}})$, where the union is over all $\mathbf{t} \in (0,1]^m$. Also, by Proposition 20, $\mathrm{SF}(\Lambda_{\mathbf{t}}) \subseteq \Lambda^{\geq}_{\mathbf{t}}$, and thus, $\mathrm{SF}(\Lambda_{\mathbf{t}}) \subseteq \Lambda^{\geq}$, by Lemma 12. Therefore, $\mathrm{SIF}(\Lambda) \subseteq \Lambda^{\geq}$.

Now suppose that $u \in \mathrm{SIF}(\Lambda)$; as shown above, we then have $u \in \Lambda^{\geq}$. By definition of $\mathrm{SIF}(\Lambda)$ there exists $\mathbf{t} \in (0,1]^m$ such that $u \in \mathrm{SF}(\Lambda_{\mathbf{t}})$. Theorem 40 implies that there exists $\mu \in I\!R^n$, that agrees on signs with u, such that $\mu \in co(\{\mathbf{t}_i\lambda_i \in \Lambda_{\mathbf{t}} : \mathbf{t}_i\lambda_i \cdot u = 1\})$, and thus, in particular, μ is in $co(\Lambda_{\mathbf{t}})$, which equals $co(\Lambda)$. Hence, there exists $\mu \in I\!R^n$, that agrees on signs with u, such that $\mu \in co(\Lambda)$.

For the converse, assume that $u \in \Lambda^{\geq}$ and there exists $\mu \in I\!R^n$, that agrees on signs with u, such that $\mu \in co(\Lambda)$. Let us define $\mathbf{t} \in I\!R^m_+$ by $\mathbf{t}(i) = \frac{1}{\lambda_i \cdot u}$ for all $i \in \{1,\ldots,m\}$. Because $u \in \Lambda^{\geq}$ we have $\lambda_i \cdot u \geq 1$, and thus, $\mathbf{t}(i) \in (0,1]$. Then, for all i we have $\mathbf{t}_i\lambda_i \cdot u = 1$, which implies that $u \in \Lambda^{\geq}_{\mathbf{t}}$ and also that $\Lambda = \{\lambda_i \in \Lambda : \mathbf{t}_i\lambda_i \cdot u = 1\}$.

Now, $co(\{\lambda_i \in \Lambda : \mathbf{t}_i\lambda_i \cdot u = 1\})$ equals $co(\{\mathbf{t}_i\lambda_i \in \Lambda_\mathbf{t} : \mathbf{t}_i\lambda_i \cdot u = 1\})$, and hence, $\mu \in co(\{\mathbf{t}_i\lambda_i \in \Lambda_\mathbf{t} : \mathbf{t}_i\lambda_i \cdot u = 1\})$. Since, $u \in \Lambda_\mathbf{t}^{\geq}$, Theorem 40 implies that $u \in \mathrm{SF}(\Lambda_\mathbf{t})$. We therefore have that $u \in \mathrm{SIF}(\Lambda)$. ■

Now, we have the following procedure to compute the $\succcurlyeq_\Lambda^{\mathrm{I,F}}$ relation; Definition 38 implies that for $\alpha, \beta \in I\!R^n$, $\alpha \not\succcurlyeq_\Lambda^{\mathrm{I,F}} \beta$ if and only if there exists $u \in \mathrm{SIF}(\Lambda)$ such that $\alpha \cdot u < \beta \cdot u$, which, by Theorem 41, is if and only if there exists $u \in I\!R^n$ and $\mu \in I\!R^n$, such that (i) $u \cdot (\beta - \alpha) > 0$, (ii) for all $i \in I$, $u \cdot \lambda_i \geq 1$ (i.e., $u \in \Lambda^{\geq}$), (iii) for all $j = \{1, \ldots, n\}$, $u(j) = 0 \iff \mu(j) = 0$, and $u(j) > 0 \iff \mu(j) > 0$ (i.e., agreeing on signs), and (iv) there exist non-negative reals r_i for each $i \in I$ such that $\mu = \sum_{i \in I} r_i \lambda_i$ (i.e., $\mu \in co(\Lambda)$).

5.6 Dealing with Inconsistencies

There are a number ways of extending the approach to deal with inconsistent input information, i.e., when $\Lambda^{\geq}$ is empty. One desirable property of such a method is that it should not depend on an arbitrary ordering of the input set Λ. Here, we describe three possible approaches for restoring consistency, which all satisfy this property.

The first approach is iteratively eliminating the elements of Λ that are least consistent with others. Define the function $C : \Lambda \to I\!R$ such that for every $i \in I$, $C(\lambda_i) = \sum_{j \in I - \{i\}} \lambda_i \cdot \lambda_j$. This function expresses a kind of degree of consistency of the element λ_i with other elements of Λ, where the smaller the value of $C(\lambda_i)$ is, the less consistency there is between λ_i and the other elements of Λ. Then, the procedure below is followed:

1. If Λ is consistent (i.e., $\Lambda^{\geq} \neq \emptyset$), return Λ.
2. Find $\gamma \in \Lambda$ that minimises C, i.e., $\gamma = \arg\min_{\lambda \in \Lambda} C(\lambda)$.
3. Remove γ from Λ, i.e., $\Lambda = \Lambda - \{\gamma\}$ and go to 1.

The second method forms a consistent subset of Λ based on the sum of the vectors. Let $\mu = \sum_{\lambda \in \Lambda} \lambda$ and define Λ_μ to be $\{\lambda \in \Lambda : \lambda \cdot \mu > 0\}$. Unless μ is the zero vector, Λ_μ is non-empty (if for all $\lambda \in \Lambda$, $\lambda \cdot \mu \leq 0$ then $(\sum_{\lambda \in \Lambda} \lambda) \cdot \mu \leq 0$, i.e., $\mu \cdot \mu \leq 0$, which only happens if $\mu = \mathbf{0}$). Also, Λ_μ is consistent because we at least have that $\mu \in (\Lambda_\mu)^{>}$, and so a positive multiple of μ is in $(\Lambda_\mu)^{\geq}$,

showing that the latter is non-empty. As a result, we can define $\omega^\star_{\Lambda_\mu}$ to be the solution of the maximum margin approach for Λ_μ. Then, we return $\Lambda_{\omega^\star_{\Lambda_\mu}} = \{\lambda \in \Lambda : \lambda \cdot \omega^\star_{\Lambda_\mu} > 0\}$ which is again consistent due to a similar reason. Here, it is evident that $\Lambda_\mu \subseteq \Lambda_{\omega^\star_{\Lambda_\mu}} \subseteq \Lambda$.

A third approach involves adding m extra real variables (i.e., m dummy features), one for each λ_i (with $i \in I = \{1, \ldots, m\}$) and extend each λ_i to the extra m variables by it having a value ε in the corresponding column, and zeros in the other $m-1$ columns. Here, ε is a strictly positive (typically small) number that relates inversely to the penalty for softening the constraints.

More formally, we say that $u \in I\!R^{n+m}$ extends $v \in I\!R^n$ if for each $j = 1, \ldots, n$, $u(j) = v(j)$. For each $i \in I$ we define δ_i as follows: δ_i extends λ_i, and $\delta_i(n+i) = \varepsilon$, and $\delta_i(n+j) = 0$ for $j \in I - \{i\}$. Let Δ, the extended preference inputs set, equal $\{\delta_i : i \in I\}$.

Consider any $w \in I\!R^n$, and any $u \in I\!R^{n+m}$ that extends w. Then, for each $i \in I$, $u \cdot \delta_i = w \cdot \lambda_i + \varepsilon u(n+i)$. Thus, $u \cdot \delta_i \geq 1$ if and only if $u(n+i) \geq \frac{1}{\varepsilon}(1 - w \cdot \lambda_i)$. If $w \cdot \lambda_i \geq 1$ then we can satisfy the constraint $u \cdot \delta_i \geq 1$ by setting $u(n+i) = 0$. Otherwise, we can satisfy the constraint by letting $u(n+i) = \frac{1}{\varepsilon}(1 - w \cdot \lambda_i)$. (In fact, since we are interested in minimising the norm, or a rescaled version of the norm, we only need to consider this particular way of extending w to $I\!R^{n+m}$.)

This implies that any $w \in I\!R^n$ can be extended to an element of $\Delta^{\geq}$; so, in particular, the extended input set Δ is always consistent. However, if w is not close to satisfying λ_i, i.e., if $w \cdot \lambda_i$ is a large negative number, then the value of $u(n+i)$, and hence the norm of u, will be large. This shows that vectors $w \in I\!R^n$ that come close to satisfying the input constraints will be favoured.

The definitions and mathematical machinery for the various preference relations defined above can then proceed as in the previous sections but now working within $I\!R^{n+m}$. When testing dominance the test vectors α and β are extended with the same value (e.g., 0) for the extra m components.

5.7 Properties of Relations and Computation of Inferences

In previous sections, we defined a number of preference relations. In Section 5.7.1 we give some properties, in particular, regarding the relationships be-

tween the preference relations. In Section 5.7.2 we express the computational characterisations, derived in earlier sections, in terms of constraints, which enable simple implementation.

5.7.1 Properties of the Different Preference Relations

We have considered the following preference relations: the consistency-based relation $\succcurlyeq_\Lambda^{\mathrm{C}}$ (Section 5.2.1), the relation $\succcurlyeq_\Lambda^{\mathrm{I}}$ based on rescaling preference inputs for the maximum margin preference relation (Section 5.3), relation $\succcurlyeq_\Lambda^{\mathrm{F}}$ based on rescaling of features (Section 5.4) and relation $\succcurlyeq_\Lambda^{\mathrm{I,F}}$ based on rescaling both inputs and features (Section 5.5).

For each of the relations $\succcurlyeq_\Lambda^{\mathrm{C}}, \succcurlyeq_\Lambda^{\mathrm{I}}, \succcurlyeq_\Lambda^{\mathrm{F}}$ and $\succcurlyeq_\Lambda^{\mathrm{I,F}}$, the corresponding *set of scenarios* is defined to be $\Lambda^{\geq}, \mathrm{SI}(\Lambda), \mathrm{SF}(\Lambda)$ and $\mathrm{SIF}(\Lambda)$, respectively. For $u \in \mathbb{R}^n$ let us define total pre-order $\geq_u$ by $\alpha \geq_u \beta \iff u \cdot \alpha \geq u \cdot \beta$. Let $\succcurlyeq$ be any of the relations $\succcurlyeq_\Lambda^{\mathrm{C}}, \succcurlyeq_\Lambda^{\mathrm{I}}, \succcurlyeq_\Lambda^{\mathrm{F}}$ and $\succcurlyeq_\Lambda^{\mathrm{I,F}}$ and let S be the corresponding set of scenarios for each relation. We then have that $\succcurlyeq$ is the intersection of relations $\geq_u$ over all $u \in S$: see Section 5.2.1, and Proposition 9, Proposition 18 and Definition 38.

The four relations, as well as $\succcurlyeq_\Lambda^{mm}$, are all reflexive and transitive, and thus pre-orders (with $\succcurlyeq_\Lambda^{mm}$ being a total pre-order). This is because each relation is equal to an intersection of pre-orders. For similar reasons, if $\succcurlyeq$ is any of the five relations then $\lambda \succcurlyeq \mathbf{0}$ for any $\lambda \in \Lambda$; and for $\alpha, \beta, \gamma \in \mathbb{R}^n$ and $r \in \mathbb{R}_+$, if $\alpha \succcurlyeq \beta$ then $\alpha + \gamma \succcurlyeq \beta + \gamma$ and $r\alpha \succcurlyeq r\beta$.

We have the following relationships between the sets of scenarios:

$\omega_\Lambda^\star \in \mathrm{SI}(\Lambda) \cap \mathrm{SF}(\Lambda)$ and $\mathrm{SI}(\Lambda) \cup \mathrm{SF}(\Lambda) \subseteq \mathrm{SIF}(\Lambda) \subseteq \Lambda^{\geq}$.

This implies the following relationships between the relations (see Figure 5.4):

$\succcurlyeq_\Lambda^{mm} \supseteq \succcurlyeq_\Lambda^{\mathrm{I}} \cup \succcurlyeq_\Lambda^{\mathrm{F}}$, and $\succcurlyeq_\Lambda^{\mathrm{I}} \cap \succcurlyeq_\Lambda^{\mathrm{F}} \supseteq \succcurlyeq_\Lambda^{\mathrm{I,F}} \supseteq \succcurlyeq_\Lambda^{\mathrm{C}}$.

5.7.2 Summary of Computational Characterisations

For finite subsets Λ of $\mathbb{R}^n$, and arbitrary $\alpha, \beta \in \mathbb{R}^n$, we would like to be able to determine which of the following hold: $\alpha \succcurlyeq_\Lambda^{\mathrm{C}} \beta$, $\alpha \succcurlyeq_\Lambda^{\mathrm{I}} \beta$, $\alpha \succcurlyeq_\Lambda^{\mathrm{F}} \beta$ and $\alpha \succcurlyeq_\Lambda^{\mathrm{I,F}} \beta$. As usual, we label Λ as $\{\lambda_i : i \in I\}$. We use the results of previous sections to express, in terms of constraints, the condition that α does not dominate β, with respect to each of the four relations.

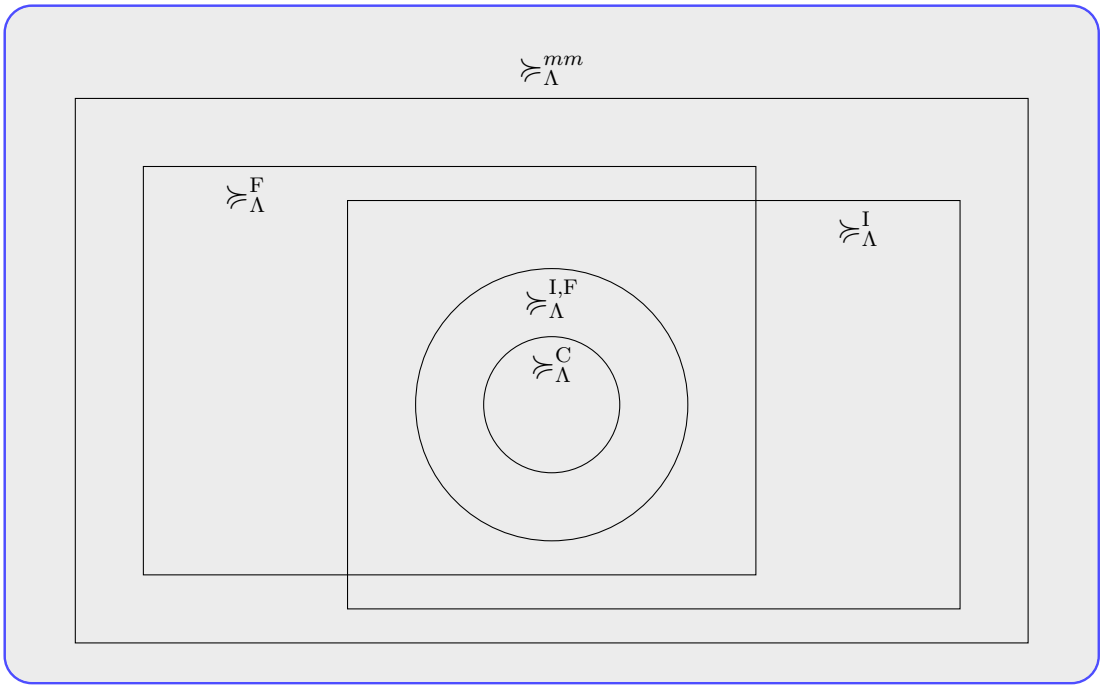

Figure 5.4: The Venn diagram that depicts relationships between the preference relations defined in this chapter.

$\succsim^{\mathbf{C}}_{\mathbf{\Lambda}}$: $\alpha \not\succsim^{\mathrm{C}}_{\Lambda} \beta$ if and only if, by Proposition 4(ii), there exists $u \in \Lambda^{\geq}$ such that $u \cdot \beta > u \cdot \alpha$. This holds if and only if there exists $u \in I\!R^n$, such that

- $u \cdot (\beta - \alpha) > 0$ and
- $\forall i \in I$, $u \cdot \lambda_i \geq 1$.

$\succsim^{\mathbf{I}}_{\mathbf{\Lambda}}$: $\alpha \not\succsim^{\mathrm{I}}_{\Lambda} \beta$ if and only if there exists $u \in \mathrm{SI}(\Lambda)$ such that $u \cdot \beta > u \cdot \alpha$. This holds, by Corollary 17, if and only if there exists $u \in I\!R^n$, and non-negative reals r_i for each $i \in I$, such that

- $u \cdot (\beta - \alpha) > 0$;
- $\forall i \in I$, $u \cdot \lambda_i \geq 1$; and
- $u = \sum_{i \in I} r_i \lambda_i$.

Note that if $\mathbf{t}$ was not restricted to $(0, 1]^m$ in the definition of $\mathrm{SI}(\Lambda)$, then the second constraint (i.e., $u \cdot \lambda_i \geq 1$) would be replaced by $u \cdot \lambda_i > 0$ which is computationally more expensive due to the strict inequality. However, as we proved in Proposition 8, the result for both cases is the same.

$\succsim^{\mathbf{F}}_{\mathbf{\Lambda}}$: $\alpha \not\succsim^{\mathrm{F}}_{\Lambda} \beta$ if and only if there exists $u \in \mathrm{SF}(\Lambda)$ such that $u \cdot \beta > u \cdot \alpha$. This holds, by Theorem 40, if and only if there exists $u \in I\!R^n$ and $\mu \in I\!R^n$, and

non-negative reals r_i for each $i \in I$, such that

- $u \cdot (\beta - \alpha) > 0$;
- $\forall i \in I$, $u \cdot \lambda_i \geq 1$;
- $\forall i \in I$, [$u \cdot \lambda_i = 1$ or $r_i = 0$];
- $\mu = \sum_{i \in I} r_i \lambda_i$;
- $\sum_{i \in I} (r_i \neq 0) \leq n + 1$; and
- $\forall j = \{1, \ldots, n\}$, $u(j) = 0 \iff \mu(j) = 0$, and $u(j) > 0 \iff \mu(j) > 0$.

In CPLEX, a disjunctive constraint such as $[w \cdot \lambda_i = 1$ or $r_i = 0]$ can be expressed as $(w \cdot \lambda_i == 1) + (r_i == 0) \geq 1$ (each logical proposition is treated as an integer; 0 for false and 1 for true).

$\succcurlyeq^{\mathrm{I,F}}_{\Lambda}$: $\alpha \not\succcurlyeq^{\mathrm{I,F}}_{\Lambda} \beta$ if and only if there exists $u \in \mathrm{SIF}(\Lambda)$ such that $u \cdot \beta > u \cdot \alpha$. This holds, by Theorem 41, if and only if there exists $u \in I\!R^n$ and $\mu \in I\!R^n$, and non-negative reals r_i for each $i \in I$, such that

- $u \cdot (\beta - \alpha) > 0$;
- $\forall i \in I$, $u \cdot \lambda_i \geq 1$;
- $\mu = \sum_{i \in I} r_i \lambda_i$; and
- $\forall j = \{1, \ldots, n\}$, $u(j) = 0 \iff \mu(j) = 0$, and $u(j) > 0 \iff \mu(j) > 0$.

5.8 Optimality Operators

In many decision-making situations, there is no clear ordering on decisions (alternatives). There can often be a set of different scenarios with a different ordering on alternatives in each scenario. For example, for different scalings of preference inputs, we may have different orderings over a set of alternatives. In such a setup there are a number of natural ways of defining the set of optimal solutions (best alternatives or top recommended solutions).

We consider here two kinds of *optimality operators* in the sense of [WRM15]; namely the set of *undominated* solutions, which is a natural generalisation of the Pareto-optimal set; and the set of *possibly optimal* solutions. The set of possibly

Algorithm 1 Finding Undominated Elements ($UND_S(A)$) Incrementally

```
1: function INCREMENTAL-UNDOMINATED(A)
2:     Ω = {}        ▷ This set contains the undominated elements found so far.
3:     for all α ∈ A do
4:         ********** Stage one **********
5:         for all ω ∈ Ω do
6:             if ω ≽ α & α ⋡ ω then                ▷ α is dominated by ω?
7:                 go to 21                          ▷ Proceed to the next α.
8:             end if
9:         end for
10:        ****************************************
11:        Reaching this point means α is not dominated by any element of Ω.
12:        Now, we will eliminate any ω in Ω that is dominated by α.
13:        ********** Stage two **********
14:        for all ω ∈ Ω do
15:            if α ≽ ω & ω ⋡ α then                ▷ α dominates ω?
16:                Ω = Ω − {ω}
17:            end if
18:        end for
19:        ****************************************
20:        Ω = Ω + {α}
21:    end for
22:    return Ω
23: end function
```

optimal alternatives has been considered in a number of different situations, including for voting rules [XC08], for soft constraint optimisation [RVW11], and for multi-objective optimisation [BP15, WRM15].

Let $\succcurlyeq$ be any of the relations $\succcurlyeq_\Lambda^{\mathrm{C}}, \succcurlyeq_\Lambda^{\mathrm{I}}, \succcurlyeq_\Lambda^{\mathrm{F}}$ and $\succcurlyeq_\Lambda^{\mathrm{I,F}}$, and let S be the corresponding set of scenarios for each relation (see Section 5.7.1), which are respectively $\Lambda^{\geq}, \mathrm{SI}(\Lambda), \mathrm{SF}(\Lambda)$ and $\mathrm{SIF}(\Lambda)$. We have then $\alpha \succcurlyeq \beta$ if and only if, for all $u \in S$, $u \cdot \alpha \geq u \cdot \beta$. We define $\succ$ to be the strict part of $\succcurlyeq$, so that $\alpha \succ \beta$ if and only if $\alpha \succcurlyeq \beta$ and $\beta \not\succcurlyeq \alpha$.

For a given set of alternatives A, the two optimality operators are defined as follows:

UND*$_S$*(A) $(= UND_\succ(A))$ is the set of undominated elements with respect to relation $\succ$, i.e., $\alpha \in UND_S(A)$ if and only if there is no $\beta \in A$ such that $\beta \succ \alpha$.

PO*$_S$*(A) is the set of elements that are optimal in some scenario. Thus, $\alpha \in PO_S(A)$ if and only if there exists $u \in S$ such that for all $\beta \in A$, $\alpha \cdot u \geq \beta \cdot u$.

From the definition of $UND_S(A)$ we have $\alpha \in UND_S(A)$ if and only if there is no $\beta \in A$ such that $\beta \succcurlyeq \alpha$ and $\alpha \not\succcurlyeq \beta$. Thus, in order to compute $UND_S(A)$ we can make use of the computation methods proposed in Section 5.7.2 for computing $\succcurlyeq$. In contrast, $PO_S(A)$ cannot be computed just from $\succcurlyeq$, because $PO_S(A)$ is not a function of $\succcurlyeq$ but rather a function of S (i.e., SF, SI etc). However, excluding the first constraint in the computation of each $\succcurlyeq$ relation (in Section 5.7.2, i.e., $\succcurlyeq^{\mathrm{I}}_{\Lambda}$, $\succcurlyeq^{\mathrm{F}}_{\Lambda}$ etc) gives a set of constraints that determines if u is in S. Therefore, we define $\mathcal{C}_S(A, \alpha)$ to be a set of constraints like $\succcurlyeq$ just by replacing the first constraint with $\forall \beta \in A$, $\alpha \cdot u \geq \beta \cdot u$. As a result, $\alpha \in PO_S(A)$ if and only if $\mathcal{C}_S(A, \alpha)$ has a solution which is if and only if there exists $u \in S$ such that $\forall \beta \in A$, $\alpha \cdot u \geq \beta \cdot u$. For example, $\mathcal{C}_{\mathrm{SI}}(A, \alpha)$ will be the following set of constraints (compare this with $\succcurlyeq^{\mathrm{I}}_{\Lambda}$ in Section 5.7.2):

- $\forall \beta \in A$, $u \cdot (\alpha - \beta) \geq 0$;
- $\forall i \in I$, $u \cdot \lambda_i \geq 1$; and
- $u = \sum_{i \in I} r_i \lambda_i$.

Typically (and as we found in our experiments), $PO_S(A)$ is a smaller set than $UND_S(A)$, although an alternative could be possibly optimal without being undominated.

Proposition 4 in [WRM15] implies that $UND_S(A)$ and $PO_S(A)$ are optimality operators, and so from Proposition 2 in [WRM15] the computation of them can be done incrementally. We exploit this for each of the four sets of scenarios.

Algorithm 1 shows how $UND_S(A)$ can be found incrementally. The algorithm consists of two stages for each $\alpha \in A$. In the first stage, we examine if α is undominated among the undominated elements Ω found so far. We proceed to the next stage if α is undominated and remove those elements of Ω that are dominated by α (so they are no longer undominated).

The set of possibly optimal elements $PO_S(A)$ is built up in an incremental way in Algorithm 2. In this algorithm, $\mathcal{F}_S(A, \alpha)$ is a function such that it returns the solution of $\mathcal{C}_S(A, \alpha)$ if a solution is found, and $NULL$ otherwise. Here, Ψ is a set of pairs where the first component of a pair is the potentially possibly optimal element, and the second one is the scenario in which the first component has been found to be optimal. Regarding this notation, $\Psi^{\downarrow}$ is the set of first components in Ψ; i.e., $\Psi^{\downarrow} = \{\psi : (\psi, u) \in \Psi\}$. In Line 6, once it is found out that α is a possibly optimal element within $\Psi^{\downarrow}$, it is included in Ψ along

Algorithm 2 Finding Possibly Optimal Elements ($PO_S(A)$) Incrementally

1: **function** INCREMENTAL-PO(A)
2: $\Psi = \{\}$
3: **for all** $\alpha \in A$ **do**
4: $u = \mathcal{F}_S(\Psi^{\downarrow}, \alpha)$ ▷ Solving for α against POs that are found so far.
5: **if** u is not $NULL$ **then** ▷ α is a PO element within $\Psi^{\downarrow}$?
6: $\Psi = \Psi + \{(\alpha, u)\}$ ▷ Keep also the solution u for later use.
7: REFINE-PREVIOUS-POS(Ψ, α)
8: **end if**
9: **end for**
10: **return** Ω
11: **end function**
12: ********
13: The following function eliminates $\psi \in \Psi^{\downarrow}$ which are no longer PO due to arrival of α.
14: ********
15: **function** REFINE-PREVIOUS-POS(Ψ, α)
16: **for all** $(\psi, v) \in \Omega$ **do**
17: **if** $\psi \cdot v < \alpha \cdot v$ **then** ▷ Is α better than ψ in the scenario v in which ψ was optimal?
18: $\Psi = \Psi - \{(\psi, v)\}$
19: $u = \mathcal{F}_S(\Psi^{\downarrow}, \psi)$ ▷ Check if there is another solution which makes ψ optimal.
20: **if** u is not $NULL$ **then**
21: $\Psi = \Psi + \{(\psi, u)\}$ ▷ Add ψ again but with the new solution u (rather than v).
22: **end if**
23: **end if**
24: **end for**
25: **end function**

with its associated solution (scenario). Then, in the function *Refine-Previous-POs*, we remove any $(\psi, v) \in \Psi$ which is not possibly optimal anymore because of adding α. In Line 18, the existing possibly optimal element ψ is removed from Ψ because it is not as good as the incoming possibly optimal element α in its own associated scenario v. However, it does not mean that ψ cannot be possibly optimal; there might be another scenario u in which ψ is better than all elements of Ψ including α. If it is the case, we include ψ again in Ψ but with this new scenario u instead of v.

5.9 Experimental Testing

The experiments make use of two databases, namely *Ridesharing Database* and *Car Preference Database*. We have explained the properties of the ridesharing

Table 5.2: The results relate to determining decisive pairs, among 1000 pairs of test vectors with respect to preference relations $\succcurlyeq_\Lambda^{\mathrm{F}}$, $\succcurlyeq_\Lambda^{\mathrm{I}}$, the intersection of $\succcurlyeq_\Lambda^{\mathrm{I}}$ and $\succcurlyeq_\Lambda^{\mathrm{F}}$ ($\succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$), $\succcurlyeq_\Lambda^{\mathrm{I,F}}$, and $\succcurlyeq_\Lambda^{\mathrm{C}}$. The bold numbers indicate that it is not always the case that $\succcurlyeq_\Lambda^{\mathrm{F}}$ is a weaker relation than $\succcurlyeq_\Lambda^{\mathrm{I}}$. The last row includes the mean of the values of each column, rounded to the nearest integer.

	m	Decisive Pairs (%)					Time (msec)			
		$\succcurlyeq_\Lambda^{\mathrm{F}}$	$\succcurlyeq_\Lambda^{\mathrm{I}}$	$\succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$	$\succcurlyeq_\Lambda^{\mathrm{I,F}}$	$\succcurlyeq_\Lambda^{\mathrm{C}}$	$\succcurlyeq_\Lambda^{\mathrm{F}}$	$\succcurlyeq_\Lambda^{\mathrm{I}}$	$\succcurlyeq_\Lambda^{\mathrm{I,F}}$	$\succcurlyeq_\Lambda^{\mathrm{C}}$
Ridesharing Database										
1.	24	21	16	9	3	1	517	36	55	18
2.	29	92	31	31	26	0.3	2434	23	40	16
3.	31	**23**	**28**	13	1	0.1	800	25	38	13
4.	36	81	35	35	31	23	4768	24	43	14
5.	38	36	19	17	5	2	2799	24	47	17
6.	41	61	12	12	12	12	5123	23	45	20
7.	53	40	20	19	19	19	1134	24	41	20
8.	55	97	26	26	24	8	1833	26	45	19
9.	62	48	24	24	11	1	4983	27	50	14
10.	94	64	35	35	5	2	5084	27	54	23
11.	127	62	24	24	24	13	6439	28	57	21
12.	129	80	36	36	19	1	2928	30	49	25
13.	134	69	28	28	28	16	7374	30	48	19
Mean	66	59	26	24	16	8	3555	27	48	19
Car Preference Database										
1.	26	65	32	31	22	10	2731	28	64	27
2.	30	42	36	28	21	12	1962	26	94	23
3.	30	36	19	17	11	7	4700	23	56	25
4.	35	56	33	30	22	9	6612	24	149	22
5.	35	65	18	17	11	5	5850	25	77	22
6.	35	**49**	**61**	41	34	20	377	26	72	31
7.	40	53	36	33	24	13	1411	56	173	78
8.	40	68	46	46	34	15	2879	26	64	25
9.	40	42	39	28	21	14	1150	26	78	27
10.	41	51	35	29	24	12	1317	28	97	23
Mean	35	53	35	30	22	12	2899	29	93	30

database in Chapter 4. The second database is the result of a survey expressing the preferences of different users over specific cars [ASBP13]. For each car 7 features are considered (e.g., engine size).

We base our experiments on 13 benchmarks derived from the ridesharing database and 10 benchmarks derived from the car preference database. Each benchmark corresponds to the inferred preferences of a different user. The preference of alternative a_i (i.e., a ridesharing alternative or a car) over b_i leads to $a_i - b_i (= \lambda_i)$ being included in Λ.

A pre-processing phase deletes some elements of Λ, in order to make it consistent (i.e., $\Lambda^{\geq} \neq \emptyset$). In order to do that, we adopt the first and the second approaches discussed in Section 5.6 respectively for the first and the second database. To conduct the experiments, CPLEX 12.6.3 is used as the solver on a computer facilitated by an Intel Xeon E312xx 2.20 GHz processor and 8 GB RAM memory.

5.9.1 Decisive Pairs

Here, we would like to examine how decisive each relation is, i.e., which relation is weaker and by how much. We randomly generate 1000 pairs (α, β), based on a uniform distribution for each feature. A pair (α, β) is called *decisive* for a preference relation if one of them can (strictly) dominate the other one; for example, the pair (α, β) is decisive for $\succcurlyeq_\Lambda^{\mathrm{I}}$ if and only if $\alpha \succ_\Lambda^I \beta$ or $\beta \succ_\Lambda^I \alpha$. This is iff either ($\alpha \succcurlyeq_\Lambda^{\mathrm{I}} \beta$ and $\beta \not\succcurlyeq_\Lambda^{\mathrm{I}} \alpha$) or ($\beta \succcurlyeq_\Lambda^{\mathrm{I}} \alpha$ and $\alpha \not\succcurlyeq_\Lambda^{\mathrm{I}} \beta$). We also consider another relation $\succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$ which is the intersection of $\succcurlyeq_\Lambda^{\mathrm{I}}$ and $\succcurlyeq_\Lambda^{\mathrm{F}}$, so that $\alpha \succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}} \beta \iff \alpha \succcurlyeq_\Lambda^{\mathrm{I}} \beta$ and $\alpha \succcurlyeq_\Lambda^{\mathrm{F}} \beta$ (note that this relation differs from the relation $\succcurlyeq_\Lambda^{\mathrm{I,F}}$).

To determine whether a pair is decisive we need to run the solver, based on the proposed computation methods in Section 5.7.2, twice; once for testing if $\alpha \succcurlyeq \beta$ and a second time for $\beta \succcurlyeq \alpha$.

Table 5.2 shows the percentage of decisive pairs for $\succcurlyeq_\Lambda^{\mathrm{F}}$, $\succcurlyeq_\Lambda^{\mathrm{I}}$, $\succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$, $\succcurlyeq_\Lambda^{\mathrm{I,F}}$ and $\succcurlyeq_\Lambda^{\mathrm{C}}$, as well as the running time per pair. Although for most of the benchmarks, $\succcurlyeq_\Lambda^{\mathrm{F}}$ is more decisive than $\succcurlyeq_\Lambda^{\mathrm{I}}$, the bold numbers show that this is not always the case. Also in some rows, like the first row, we see that $\succcurlyeq_\Lambda^{\mathrm{I}}$ is greater than $\succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$, which implies that $\succcurlyeq_\Lambda^{\mathrm{I}} \not\subseteq \succcurlyeq_\Lambda^{\mathrm{F}}$. In terms of running time, $\succcurlyeq_\Lambda^{\mathrm{I}}$ is around 130 and 100 times faster than $\succcurlyeq_\Lambda^{\mathrm{F}}$ on average for the ridesharing database and the car preference database respectively. Also, the results illustrate the fact that $\succcurlyeq_\Lambda^{\mathrm{C}} \subseteq \succcurlyeq_\Lambda^{\mathrm{I,F}} \subseteq \succcurlyeq_\Lambda^{\mathrm{I}\wedge\mathrm{F}}$.

Table 5.3: A comparison, between the number of possibly optimal elements and the number of undominated elements among 100 alternatives with regard to preference relations $\succcurlyeq_{\Lambda}^{\mathrm{C}}$, $\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$, $\succcurlyeq_{\Lambda}^{\mathrm{I}}$, and $\succcurlyeq_{\Lambda}^{\mathrm{F}}$. The $\mathrm{I} \cap \mathrm{F}$ column relates to the intersection of the I and F columns. The bold numbers illustrate that the F and $\mathrm{I} \cap \mathrm{F}$ sets are not always identical (so that the F optimality set is not always a subset of the I optimality set), and the encircled numbers relate to the cases when $|PO_S(A)| > |UND_S(A)|$. The last row includes the mean of values of each column, rounded to the nearest integer.

	$\lvert PO_S(A)\rvert$					$\lvert UND_S(A)\rvert$				
	C	I, F	I	F	I ∩ F	C	I, F	I	F	I ∩ F
Ridesharing Database										
1.	38	26	20	**6**	**4**	72	55	33	**16**	**13**
2.	45	13	12	2	2	86	20	15	3	3
3.	64	37	21	**6**	**5**	97	74	30	**19**	**18**
4.	7	7	7	3	3	7	7	7	4	4
5.	33	32	21	**13**	**12**	63	54	38	17	17
6.	14	14	14	5	5	18	18	18	5	5
7.	10	10	10	6	6	18	18	17	7	7
8.	18	9	9	1	1	25	12	12	1	1
9.	34	17	13	6	6	78	19	15	8	8
10.	22	15	8	2	2	50	38	13	2	2
11.	20	14	14	2	2	27	19	19	3	3
12.	41	12	9	2	2	79	24	15	2	2
13.	16	12	12	4	4	29	16	16	6	6
Mean	28	17	13	4	4	50	29	19	7	7
Car Preference Database										
1.	14	14	11	**6**	**5**	18	18	12	6	6
2.	14	10	3	**5**	**2**	15	11	4	**5**	**3**
3.	34	30	16	**7**	**3**	36	31	26	12	12
4.	17	11	8	2	2	20	13	11	3	3
5.	29	22	21	(11)	(11)	34	26	21	(**9**)	(**7**)
6.	8	5	3	2	2	8	5	3	2	2
7.	12	6	5	3	3	12	6	5	3	3
8.	14	11	5	**4**	**2**	15	13	8	**4**	**2**
9.	14	11	7	(**5**)	(**4**)	14	11	7	(**4**)	(**3**)
10.	15	10	9	4	4	16	10	9	4	4
Mean	17	13	9	5	4	19	14	11	5	5

Table 5.4: A comparison, between the running time for finding possibly optimal elements and undominated elements among 100 alternatives with regard to preference relations $\succcurlyeq_{\Lambda}^{\mathrm{C}}$, $\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$, $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ and $\succcurlyeq_{\Lambda}^{\mathrm{F}}$. The last row includes the mean of values of each column, rounded to the nearest integer.

	$PO_S(A)$ Time (s)				$UND_S(A)$ Time (s)			
	C	I, F	I	F	C	I, F	I	F
Ridesharing Database								
1.	31	53	18	66	215	187	97	128
2.	41	39	22	505	152	46	24	516
3.	37	103	17	234	176	241	43	570
4.	7	11	13	29	9	18	10	870
5.	13	29	22	572	124	166	68	1710
6.	17	22	21	277	32	53	29	1723
7.	11	14	17	86	24	32	21	259
8.	13	16	8	334	42	20	12	243
9.	26	34	9	405	162	42	23	1729
10.	27	31	14	654	151	136	25	1147
11.	15	27	13	412	51	48	33	1835
12.	41	23	14	330	272	46	22	558
13.	27	24	19	539	68	46	29	2087
Mean	24	33	16	342	114	83	34	1029
Car Preference Database								
1.	14	62	21	226	37	88	19	471
2.	10	47	8	185	25	38	11	870
3.	18	33	13	407	51	91	53	2344
4.	10	51	15	338	19	42	13	2126
5.	22	35	17	603	43	156	33	1535
6.	11	30	13	40	35	44	53	367
7.	12	48	20	98	63	63	55	275
8.	10	52	10	294	20	22	8	415
9.	10	24	13	55	22	50	6	72
10.	7	34	10	149	26	41	9	374
Mean	12	42	14	240	34	64	26	885

5.9.2 Optimal Elements

The next phase of experiments is devoted to finding optimal solutions with respect to the two kinds of optimality operator discussed in Section 5.8. To do so, a set of 100 alternatives (i.e., the set A) is randomly generated, based on a uniform distribution for each feature. Then, for each relation, the number of possibly optimal and undominated elements in A is counted; see Table 5.3. The numbers in the $\mathrm{I} \cap \mathrm{F}$ columns relate to the intersection of the I and F optimality sets; for example, the left-hand $\mathrm{I} \cap \mathrm{F}$ column gives the cardinalities of the sets $PO_{\mathrm{SI}(\Lambda)} \cap PO_{\mathrm{SF}(\Lambda)}$. The bold numbers show that the F and $\mathrm{I} \cap \mathrm{F}$ columns are not identical, and thus illustrate that e.g., $PO_{\mathrm{SF}(\Lambda)}$ is not necessarily a subset of $PO_{\mathrm{SI}(\Lambda)}$. In the ridesharing database, it can be seen that for the most conservative relation, $\succcurlyeq_{\Lambda}^{\mathrm{C}}$, the optimality operators return a substantial proportion of alternatives as optimal solutions (roughly half for $UND_S(A)$). Moreover, the number of undominated elements for the car preference database—unlike the ridesharing database—is fairly similar to the number of possibly optimal elements, and we sometimes even have $|PO_S(A)| > |UND_S(A)|$ (see the encircled numbers).

Table 5.4 shows the time for finding possibly optimal and undominated solutions, where the former is faster than the latter by a factor ranging from 1.5 to 4.8 on average; this is partly because of $|PO_S(A)|$ being usually smaller than $|UND_S(A)|$ particularly for the ridesharing database. Because the computation of $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ was very much slower than the other relations, the times in the F columns are still greatest, despite the number of optimal solutions being smaller. Overall, the computational cost of the $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ may make it less useful, even though it is more decisive, and thus leads to smaller sets of optimal solutions. Instead one might, for instance, favour $PO_{\mathrm{SI}(\Lambda)}$, $PO_{\mathrm{SIF}(\Lambda)}$ and $UND_{\mathrm{SI}(\Lambda)}$ since they generate reasonably sized optimality sets much faster.

5.10 Summary and Discussion

The maximum margin method for preference learning learns a utility function from a set of input preferences, in order to predict further preferences. However, in many situations, it can be argued that the scaling of preference inputs should not affect the induced preference relation. We have defined a relation $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ that is a more robust version of the maximum margin preference infer-

ence $\succcurlyeq_{\Lambda}^{mm}$, and which is invariant to the scaling of preference inputs. It is also reasonable to consider invariance to the way that features are scaled because, in maximum margin inference, features should be scaled before applying the method; this is due to the fact that the objective function in maximum margin method is sensitive to the scale of feature domains. Thus, we have also defined the $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ relation which is invariant to the scaling of features. With these two types of rescaling being complementary, it is also natural to consider both types simultaneously, leading to a further preference relation $\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$. We derived characterisations for the relations $\succcurlyeq_{\Lambda}^{\mathrm{I}}$, $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ and $\succcurlyeq_{\Lambda}^{\mathrm{I,F}}$, which lead to computational procedures. We also characterised the situation when the maximum margin relation is insensitive to the scaling of features, i.e., $\succcurlyeq_{\Lambda}^{\mathrm{F}}$ equals $\succcurlyeq_{\Lambda}^{mm}$. We then discussed three basic approaches to restore consistency of input data. Two optimality operators—$UND_S(A)$ and $PO_S(A)$—have been considered to define how a set of optimal solutions can be extracted from the available alternatives. We proposed two algorithms in order to compute $UND_S(A)$ and $PO_S(A)$ in an incremental manner. Our experiments, which used 23 benchmarks derived from two sets of real preference data, compared the different relations in terms of decisiveness and the set of optimal solutions regarding $UND_S(A)$ and $PO_S(A)$, and showed that the computational methods are practically feasible for a moderate number of instances/features. The relation associated with only scaling the features was the most decisive but by far the slowest for computing the associated optimality classes. Overall, one might consider $\succcurlyeq_{\Lambda}^{\mathrm{I}}$ as a relation that keeps quite a good balance between decisiveness and computation time.

In the future, it would be interesting to explore extensions of our approaches including (i) integration of the approach with a conversational recommender system; (ii) developing computational methods for certain kinds of kernel; (iii) considering soft margin optimisation, i.e., more sophisticated approaches for dealing with an inconsistent dataset; (iv) taking into account more general kinds of input preference statement; and (v) exploring connections with imprecise probability, based on linear constraints on probabilities.

Appendix of Chapter 5

The appendix includes all the proofs, of the results in this chapter, that do not appear in the main body of the paper.

Lemma 2. *Consider any $\Lambda \subseteq \mathbb{R}^n$. If $\Lambda^>$ is non-empty then Λ^* is the topological closure of $\Lambda^>$.*

Proof: Let us write the topological closure operator as $Cl(\cdot)$, so that $Cl(S)$ is the topological closure of S, which equals S plus all the limit points of S. Basic properties of $Cl(\cdot)$ include: (a) $S \subseteq T$ implies $Cl(S) \subseteq Cl(T)$, and (b) $Cl(S) = S$ if S is a topologically closed set.

It is clear that $\Lambda^> \subseteq \Lambda^*$ which implies that $Cl(\Lambda^>) \subseteq Cl(\Lambda^*)$; also, $Cl(\Lambda^*) = \Lambda^*$ since Λ^* is a topologically closed set. We thus have $Cl(\Lambda^>) \subseteq \Lambda^*$.

Now, we will show that $\Lambda^* \subseteq Cl(\Lambda^>)$. To do so, we will prove that for any $x \in \Lambda^*$ there is a sequence of elements of $\Lambda^>$ that converges to x. Choose arbitrary $x \in \Lambda^*$ and $y \in \Lambda^>$. For each $\delta \in (0,1)$, and each $\lambda \in \Lambda$, $x \cdot \lambda \geq 0$ and $y \cdot \lambda > 0$, because $x \in \Lambda^*$ and $y \in \Lambda^>$, and thus, $(\delta x + (1-\delta)y) \cdot \lambda = \delta(x \cdot \lambda) + (1-\delta)(y \cdot \lambda) > 0$, and so, $\delta x + (1-\delta)y \in \Lambda^>$. As δ tends to 1, $\delta x + (1-\delta)y$ tends to x, showing that $x \in Cl(\Lambda^>)$, as required. □

Lemma 3. *Consider any finite $\Lambda \subseteq \mathbb{R}^n$ and any $u \in \mathbb{R}^n$. Then, $\Lambda^* \subseteq \{u\}^*$ if and only if $u \in co(\Lambda)$.*

Proof: Because $\Lambda^* = (co(\Lambda))^*$ we have that $\Lambda^* \subseteq (\{u\})^*$ if and only if $(co(\Lambda))^* \subseteq (co(\{u\}))^*$. Now, clearly, if $u \in co(\Lambda)$ then $(co(\Lambda))^* \subseteq (\{u\})^*$, and thus $\Lambda^* \subseteq (\{u\})^*$. To prove the converse, it is sufficient to show that $(co(\Lambda))^* \subseteq (co(\{u\}))^*$ implies $u \in co(\Lambda)$. Now, $(co(\Lambda))^* \subseteq (co(\{u\}))^*$ implies $(co(\Lambda))^{**} \supseteq (co(\{u\}))^{**}$. Convex cones $co(\Lambda)$ and $co(\{u\})$ are both topologically closed (because Λ is finite), so, by a fundamental result for convex cones (see e.g., [BV04, Section 2.6.1]) $(co(\Lambda))^{**} = co(\Lambda)$ and $(co(\{u\}))^{**} = co(\{u\})$, and thus $co(\{u\}) \subseteq co(\Lambda)$, which implies that $u \in co(\Lambda)$. □

Lemma 6. *Consider any finite $\Lambda \subseteq \mathbb{R}^n$, any $\mathbf{t} \in \mathbb{R}^m_+$, any $r \in \mathbb{R}_+$, and any $v \in \mathbb{R}^n$. If $\mathbf{t}' = \frac{t}{r}$ then the following results hold.*

(i) $v \in \Lambda^{\geq}_{\mathbf{t}}$ if and only if rv is in $\Lambda^{\geq}_{\mathbf{t}'}$.

(ii) $\omega^{\star}_{\Lambda_{\mathbf{t}'}} = r\omega^{\star}_{\Lambda_{\mathbf{t}}}$; i.e., v has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}}$ if and only if rv has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}'}$.

Proof: (i): $v \in \Lambda^{\geq}_{\mathbf{t}}$ if and only if for all $i \in I$, $v \cdot (\mathbf{t}_i \lambda_i) \geq 1$, which is if and only if for all $i \in I$, $(\frac{\mathbf{t}_i}{r} rv) \cdot \lambda_i \geq 1$, which holds if and only if for all $i \in I$, $rv \cdot (\mathbf{t}'_i \lambda_i) \geq 1$,

which is iff $rv \in \Lambda^{\geq}_{\mathbf{t}'}$.

(ii): v has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}}$ if and only if $v \in \Lambda^{\geq}_{\mathbf{t}}$ and for all $u \in \Lambda^{\geq}_{\mathbf{t}}$, $\|u\| \geq \|v\|$. Part (i) tells us that $v \in \Lambda^{\geq}_{\mathbf{t}} \iff rv \in \Lambda^{\geq}_{\mathbf{t}'}$. Now, for all $u \in \Lambda^{\geq}_{\mathbf{t}}$, $\|u\| \geq \|v\|$ holds if and only if for all $u \in \Lambda^{\geq}_{\mathbf{t}}$, $\|ru\| \geq \|rv\|$ which, from (i), is if and only if for all $ru \in \Lambda^{\geq}_{\mathbf{t}'}$, $\|ru\| \geq \|rv\|$, i.e., for all $u' \in \Lambda^{\geq}_{\mathbf{t}'}$, $\|u'\| \geq \|rv\|$. Thus, v has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}}$ if and only if $v \in \Lambda^{\geq}_{\mathbf{t}}$ and for all $u' \in \Lambda^{\geq}_{\mathbf{t}'}$, $\|u'\| \geq \|rv\|$. This holds if and only if rv has the minimum norm in $\Lambda^{\geq}_{\mathbf{t}'}$. □

Lemma 14. *Consider any $u \in G$ where $G \subseteq \mathbb{R}^n$ is a convex set. Then, u has the minimum norm in G if and only if for all $v \in G$, $u \cdot (v - u) \geq 0$.*

Proof: $\Rightarrow$: Firstly, for the case when $v = u$, the result is easily obtained because $u \cdot (v - u) = 0$. Now, consider any $v \in G \setminus \{u\}$. We define $v_\delta = \delta v + (1 - \delta)u$ for each $\delta \in (0, 1]$. It is clear that $v_\delta \in G$ because v and u both are in the convex set G, and since u has the minimum norm in G, for all $\delta \in (0, 1]$ we have that $\|v_\delta\| \geq \|u\|$. Now, assume that $u \cdot (v - u) < 0$. We show that this assumption leads to $\|v_\delta\| < \|u\|$ for some $\delta \in (0, 1]$, which will prove the first part by contradiction. To do this, we rewrite $\|v_\delta\|^2 - \|u\|^2$ as follows:

$$\begin{aligned}\|v_\delta\|^2 - \|u\|^2 &= \|\delta(v - u) + u)\|^2 - \|u\|^2 \\ &= (\delta(v - u) + u)) \cdot (\delta(v - u) + u)) - u \cdot u,\end{aligned}$$

which equals $\delta^2(v - u) \cdot (v - u) + 2\delta u \cdot (v - u)$, i.e., $\delta(\delta\|v - u\|^2 + 2u \cdot (v - u))$. Now, since $u \cdot (v - u) < 0$, for sufficiently small δ, $\|v_\delta\|^2 - \|u\|^2 < 0$, and thus $\|v_\delta\| < \|u\|$.

$\Leftarrow$: Consider any $v \in G \setminus \{u\}$. Since $u \neq v$, $\|v - u\|^2 > 0$, which implies that $(v - u) \cdot (v - u) > 0$, and thus, $\|v\|^2 + \|u\|^2 > 2v \cdot u$. Also, $u \cdot (v - u) \geq 0$ leads to $v \cdot u \geq \|u\|^2$. Hence, $\|v\|^2 + \|u\|^2 > 2\|u\|^2$, and thus, $\|v\| > \|u\|$, showing that u has minimum norm in G. □

Lemma 23. *Let $u, v \in \mathbb{R}^n$. There exists $k \in \{1, \ldots, n\}$ such that $|u(k)| < |v(k)|$ if and only if there exists $\tau \in \mathbb{R}^n_+$ such that $\|u \odot \tau\| < \|v \odot \tau\|$. Thus, for all $j \in \{1, \ldots, n\}$, $|u(j)| \geq |v(j)|$ if and only if for all $\tau \in \mathbb{R}^n_+$, $\|u \odot \tau\| \geq \|v \odot \tau\|$.*

Proof: $\Rightarrow$: Assume first that there exists $k \in \{1, \ldots, n\}$ such that $|u(k)| < |v(k)|$. For $\epsilon > 0$, define $\tau_\epsilon \in \mathbb{R}^n_+$ by $\tau_\epsilon(k) = 1 + \epsilon$, and, for $j \neq k$, $\tau_\epsilon(j) = \epsilon$. Then $u \odot \tau_\epsilon = \epsilon u + u(k)e_k$, where e_k is the unit vector in the kth direction, which

leads to $\|u \odot \tau_\epsilon\|^2 = (u \odot \tau_\epsilon) \cdot (u \odot \tau_\epsilon)$ equalling $\epsilon^2 u \cdot u + (1+2\epsilon)u(k)^2$. Similarly, $\|v \odot \tau_\epsilon\|^2 = \epsilon^2 v \cdot v + (1+2\epsilon)v(k)^2$. Since $u(k)^2 < v(k)^2$, for sufficiently small $\epsilon > 0$, we will have $\|u \odot \tau_\epsilon\|^2 < \|v \odot \tau_\epsilon\|^2$.

$\Leftarrow$: Now assume that there exists $\tau \in \mathbb{R}^n_+$ such that $\|u \odot \tau\| < \|v \odot \tau\|$. Then for some $k \in \{1, \ldots, n\}$, $|(u \odot \tau)(k)| < |(v \odot \tau)(k)|$, i.e., $|u(k)\tau(k)| < |v(k)\tau(k)|$, which implies $|u(k)| < |v(k)|$, since $\tau(k)$ is non-zero. □

Lemma 24. *Let G be a convex subset of $\mathbb{R}^n$, and let j be any element of $\{1, \ldots, n\}$. Then either*

(i) there exists $w \in G$ such that $w(j) = 0$; or

(ii) for all $w \in G$, $w(j) > 0$; or

(iii) for all $w \in G$, $w(j) < 0$.

Proof: To prove a contradiction, suppose that neither (i), (ii) nor (iii) hold for j, so for all $w \in G$, $w(j) \neq 0$, and there exists $u, v \in G$ such that $u(j) > 0$ and $v(j) < 0$. Let $\delta = \frac{u(j)}{u(j)-v(j)}$, so that $1 - \delta = \frac{-v(j)}{u(j)-v(j)}$. Let $v_\delta = \delta v + (1-\delta)u$, which is in G because G is convex and $\delta \in (0, 1)$. Then, $v_\delta(j) = 0$, which shows that (i) holds for j, contradicting the earlier assumption. □

Lemma 28. *Consider any convex set $G \in \mathbb{R}^n$. Then, u is zm-pointwise undominated in convex G if and only if for all $v \in G$, either*

(i) $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$; or

(ii) there exists $k \in \{1, \ldots, n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$.

Proof: First, let us suppose that u is not zm-pointwise undominated in G. We will show that there exists $v \in G$ such that neither condition (i) nor condition (ii) hold for v. Since u is not zm-pointwise undominated in G, there exists $v \in G$ that zm-pointwise dominates u. By definition, there exists $j \in \{1, \ldots, n\}$ such that $v(j) \neq u(j) \neq 0$, and thus, condition (i) does not hold for v; also for all $k \in \{1, \ldots, n\}$ with $u(k) \neq 0$, either $0 \leq v(k) \leq u(k)$ or $0 \geq v(k) \geq u(k)$, which means that condition (ii) in this lemma does not hold for v.

Conversely, suppose that it is not the case that for all $v \in G$, either (i) $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$; or (ii) there exists $k \in \{1, \ldots, n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$. Then, there exists $v \in G$

such that (i) there exists $k \in \{1,\ldots,n\}$ such that $u(k) \neq 0$ and $u(k) \neq v(k)$; and (ii) for all $j \in \{1,\ldots,n\}$, if $u(j) > 0$ then $v(j) \leq u(j)$; and if $u(j) < 0$ then $v(j) \geq u(j)$. Thus, there exists $v \in G$ such that (i) there exists $k \in N_u$ such that $u(k) \neq v(k)$; and (ii) for all $j \in N_u$, if $u(j) > 0$ then $v(j) \leq u(j)$; and if $u(j) < 0$ then $v(j) \geq u(j)$ (where $N_u = \{j \in \{1,\ldots,n\} : u(j) \neq 0\}$, as in Definition 36). For $\delta \in (0,1]$ let $v_\delta = \delta v + (1-\delta)u$, which is in G. Then there exists $\delta \in (0,1]$ such that (i) there exists $k \in N_u$ such that $u(k) \neq v_\delta(k)$; and (ii) for all $j \in N_u$, if $u(j) > 0$ then $0 < v_\delta(j) \leq u(j)$; and if $u(j) < 0$ then $0 > v_\delta(j) \geq u(j)$. Thus, v_δ zm-pointwise dominates u showing that u is not pointwise undominated in G. □

Lemma 29. *Let $u, v \in \mathbb{R}^n$, with $u \neq v$, and let $\tau \in \mathbb{R}^n_+$. For $\delta \in (0,1]$ let $v_\delta = \delta v + (1-\delta)u$. Then the following hold:*

(i) For any $\delta \in (0,1]$, $\|v_\delta \odot \tau\|^2 - \|u \odot \tau\|^2 = \delta^2\|(v-u) \odot \tau\|^2 + 2\delta(\tau \odot \tau \odot u) \cdot (v-u)$.

(ii) $(\tau \odot \tau \odot u) \cdot (v-u) \geq 0$ if and only if for all $\delta \in (0,1]$, $\|v_\delta \odot \tau\| > \|u \odot \tau\|$.

(iii) There exists $\tau \in \mathbb{R}^n_+$ such that $(\tau \odot \tau \odot u) \cdot (v-u) \geq 0$ if and only if either (a) $v(j) = u(j)$ for all $j \in \{1,\ldots,n\}$ such that $u(j) \neq 0$; or (b) there exists $k \in \{1,\ldots,n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$.

Proof: (i): Using $v_\delta = u + \delta(v-u)$, we have that $v_\delta \odot \tau = (u \odot \tau) + \delta(v-u) \odot \tau$. Then, $\|v_\delta \odot \tau\|^2 = (v_\delta \odot \tau) \cdot (v_\delta \odot \tau) = (u \odot \tau) \cdot (u \odot \tau) + \delta^2\|(v-u) \odot \tau\|^2 + 2\delta(u \odot \tau) \cdot ((v-u) \odot \tau)$, which leads to the result.

(ii): If $(\tau \odot \tau \odot u) \cdot (v-u) \geq 0$ then (i) immediately implies that for all $\delta \in (0,1]$, $\|v_\delta \odot \tau\| > \|u \odot \tau\|$, since $\|(v-u) \odot \tau\|$ is non-zero (because $u \neq v$). Conversely, suppose that $(\tau \odot \tau \odot u) \cdot (v-u) < 0$. Choosing δ such that $\delta\|(v-u) \odot \tau\|^2 \leq -2(\tau \odot \tau \odot u) \cdot (v-u)$ gives, using (i), that $\|v_\delta \odot \tau\| \leq \|u \odot \tau\|$, proving (ii).

(iii), $\Rightarrow$: Suppose that there exists $\tau \in \mathbb{R}^n_+$ such that $(\tau \odot \tau \odot u) \cdot (v-u) \geq 0$, and it is not the case that there exists $k \in \{1,\ldots,n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$. Then, we can see that for each $j \in \{1,\ldots,n\}$, $(\tau \odot \tau \odot u)(j)(v-u)(j) = \tau(j)^2 u(j)(v(j)-u(j)) \leq 0$ (since it clearly holds if $u(j) = 0$; if $u(j) > 0$ then $v(j) \leq u(j)$ so it also holds; if $u(j) < 0$ then $v(j) \geq u(j)$ so it holds then too). The sum (over each j) of these n terms is at least zero, since it is equal to $(\tau \odot \tau \odot u) \cdot (v-u)$ and thus, for each $j \in \{1,\ldots,n\}$,

$(\tau \odot \tau \odot u)(j)(v-u)(j) = 0$. This implies that $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$.

$\Leftarrow$: If (a) $v(j) = u(j)$ for all $j \in \{1, \ldots, n\}$ such that $u(j) \neq 0$ then $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$. Now, assume that (b) holds, i.e., there exists $k \in \{1, \ldots, n\}$ such that either $0 < u(k) < v(k)$ or $0 > u(k) > v(k)$. For $\epsilon > 0$, define τ_ϵ by $\tau_\epsilon(k) = \sqrt{1+\epsilon}$, and $\tau_\epsilon(j) = \sqrt{\epsilon}$ for all $j \neq k$. Then, $\tau_\epsilon \odot \tau_\epsilon \odot u = u(k)e_k + \epsilon u$, where e_k is the unit vector in the kth direction, so $(\tau_\epsilon \odot \tau_\epsilon \odot u) \cdot (v-u) = u(k)(v(k)-u(k)) + \epsilon u \cdot (v-u)$, which is greater than zero for sufficiently small ϵ, since $u(k)(v(k) - u(k)) > 0$.□

Lemma 32. *Consider any $u \in G_I$ and any $v \in G_{J_u}$. Then there exists $\delta' \in (0, 1)$ such that for all δ with $0 < \delta \leq \delta'$, $\delta v + (1-\delta)u \in G_I$.*

Proof: Let $x = v - u$, and, for all $\delta \in (0, 1)$, let $v_\delta = u + \delta x = \delta v + (1-\delta)u$. Since $u, v \in G_{J_u}$, we have $v_\delta \in G_{J_u}$ for all $\delta \in (0, 1)$. We will next show that for all sufficiently small δ, $v_\delta \in G_I$, i.e., that for all $i \in I$, $v_\delta \cdot \lambda_i \geq a_i$. Since, $v_\delta \in G_{J_u}$, this holds for all $i \in J_u$. Now, consider any $i \in I \setminus J_u$. By definition of J_u we have $u \cdot \lambda_i > a_i$. This implies that there exists $\delta_i > 0$ with for all δ with $0 < \delta \leq \delta_i$, $(u \cdot \lambda_i) + \delta(x \cdot \lambda_i) > a_i$, and thus $v_\delta \cdot \lambda_i > a_i$. Let us choose $\delta' = \min\{\delta_i : i \in I \setminus J_u\}$. Then for all δ with $0 < \delta \leq \delta'$, and for all $i \in I \setminus J_u$, $v_\delta \cdot \lambda_i > a_i$, so for all $i \in I$, $v_\delta \cdot \lambda_i \geq a_i$, which shows that $v_\delta \in G_I$. □

Lemma 33. *Consider non-zero $u \in G_I$ (as defined above). Then u is zm-pointwise undominated in G_I if and only if u is zm-pointwise undominated in G_{J_u}.*

Proof: $\Rightarrow$: Suppose that u is zm-pointwise undominated in G_I. Consider any $v \in G_{J_u}$. By Lemma 32, there exists $\delta \in (0, 1)$ such that $v_\delta \in G_I$, where $v_\delta = \delta v + (1-\delta)u = u + \delta(v-u)$. Proposition 30(ii) implies that there exists $\tau \in \mathbb{R}^n_+$ such that for all $w \in G_I$, $(\tau \odot \tau \odot u) \cdot (w - u) \geq 0$. In particular, $(\tau \odot \tau \odot u) \cdot (v_\delta - u) \geq 0$, i.e., $(\tau \odot \tau \odot u) \cdot (\delta(v-u)) \geq 0$, which implies that $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$. Note that δ does not depend on the choice of v. Thus, there exists $\tau \in \mathbb{R}^n_+$ such that for all $v \in G_{J_u}$, $(\tau \odot \tau \odot u) \cdot (v - u) \geq 0$. Applying Proposition 30(ii) again gives that u is zm-pointwise undominated in G_{J_u}.

$\Leftarrow$: This is immediate because $G_I \subseteq G_{J_u}$. □

Lemma 35. *For $u, v \in \mathbb{R}^n$, If u and v agree on signs and $u \neq \mathbf{0}$ then $u \cdot v > 0$.*

Proof: Because $u \neq \mathbf{0}$, there exists $k \in \{1, \ldots, n\}$ with $u(k) \neq 0$. Then, since u and v agree on signs, $v(k)$ is non-zero and the same sign as $u(k)$, so $u(k)v(k) > 0$. Also, for any $j \in \{1, \ldots, n\}$, $u(j)v(j) \geq 0$, and thus, $u \cdot v > 0$. □

Lemma 37. *Consider a polyhedron G_I and non-zero $u \in G_I$. Then u is rescale-optimal in G_I if and only if u is rescale-optimal in G_{J_u}.*

Proof: Firstly, since $G_I \subseteq G_{J_u}$, if u is rescale-optimal in G_{J_u} then u is rescale-optimal in G_I (since the same scaling function τ can be used). We will go on to prove the converse; so, let us assume that u is rescale-optimal in G_I. Theorem 31 implies that there exists $\mu \in I\!R^n$ agreeing on signs with u such that $\mu \cdot u = 1$ and for all $w \in G_I$, $\mu \cdot w \geq 1$. Consider arbitrary $v \in G_{J_u}$; we will show that $\mu \cdot v \geq 1$.

Let $x = v - u$, and, for all $\delta \in (0, 1)$, let $v_\delta = \delta v + (1 - \delta)u = u + \delta x$. By Lemma 32, there exists $\delta \in (0, 1)$ such that $v_\delta \in G_I$. This implies that $\mu \cdot v_\delta \geq 1$, so $\mu \cdot u + \delta\mu \cdot x \geq 1$, and hence $\delta\mu \cdot x \geq 0$ and $\mu \cdot x \geq 0$, and therefore, $\mu \cdot v \geq \mu \cdot u = 1$.

We have shown that for all $v \in G_{J_u}$, $\mu \cdot v \geq 1$; we also have that μ and u agree on signs and $\mu \cdot u = 1$. Using Theorem 31, this implies that u is rescale-optimal in G_{J_u}, as required. □

Chapter 6

Rescale-Invariant SVM for Binary Classification

6.1 Introduction

In the previous chapter, we discussed that the maximum margin preference learning method is sensitive to the way that features are scaled. This is also the case for SVM [Bur98] as one of the best-known classification approaches; i.e., different ways of scaling of features may lead to different classification results. Considering the fact that rescaling (normalisation) of features is a necessary pre-processing phase for SVM, it is therefore natural to consider a more cautious classification in which an instance is *strongly* positively (negatively) classified if it is labelled as positive (negative) for all choices of scaling; the other instances, whose classification can depend on the choice of scaling, are labelled as neutral. Thus, this method refines the set of positively (negatively) classified instances in order to improve the level of confidence in the classification decisions. This could be helpful in certain sensitive decision-making applications such as disease diagnosis; e.g., the test for presence of a particular disease would fall into three categories, *positive*, *negative*, and *requires further examination*.

In this chapter, we derive a way of characterising the approach for binary SVM that allows determining when an instance strongly belongs to a class and when the classification is invariant to rescaling. The characterisation leads to a computational method to determine whether one sample is strongly positive, strongly negative or neither. Our experimental results back up the intuition that being strongly positive suggests stronger confidence that an instance really is positive.

There are some studies attempting to account for the dependence on feature scaling in margin-based optimisation methods from a different perspective; for example, authors in [JS09] try to deal with this problem by maximizing the margin relative to the spread of data, or another approach that is discussed in [DS03] considers a selection of features so that the classifier will be scaling-invariant with respect to those features. Also, note that the motivation behind this work is different from studies, like [XCS06], which are concerned with improving the robustness of SVM against outliers and noise, since we are specifically focusing on the uncertainty caused by rescaling of features.

The rest of this chapter is organised as follows. In Section 6.2, we extend the terminology of conventional binary SVM to facilitate our results. Section 6.3 considers the effect of rescaling and defines strong classification. In Section 6.4,

we characterise strong classification using the concept of rescale-optimality, where the rescale-optimal vectors are those that can be made optimal in the SVM objective function for some rescaling. This characterisation leads to a method for computing strong classification, as described in Section 6.5. In Section 6.6, the presented approach is evaluated with 18 benchmarks, which are derived from six real data sets.

6.2 Standard SVM for Binary Classification

In this section, we introduce some notation, along with some relevant results for standard SVM. This enables easy generalisation to the rescale-invariant case. In this chapter, as an initial step, we just consider the case when the training set is consistent; i.e., the instances are linearly separable. However, we can take similar approaches described in Section 5.6 to deal with inconsistencies.

Recall the formalism that was used in Section 2.3 for conventional SVM, where $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ is the set of training samples, a sample is expressed as a pair of (x, y), $X^+ = \{x^+ : (x^+, +1) \in X\}$, and $X^- = \{x^- : (x^-, -1) \in X\}$.

We say that X is *non-trivial* if both X^+ and X^- are non-empty. We define the following terms for any $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ which are used throughout the chapter, and illustrated in Example 17 below.

- $\Lambda(X) = \{\frac{x^+ - x^-}{2} : x^+ \in X^+,\ x^- \in X^-\}$
- $I(X) = \{1, \ldots, |X^+||X^-|\}$
- $G(X) = \{w \in \mathbb{R}^n : \forall \lambda \in \Lambda(X),\ w \cdot \lambda \geq 1\}$
- $MP_w = \min_{x^+ \in X^+} w \cdot x^+$
- $MN_w = \min_{x^- \in X^-} w \cdot (-x^-)$

Example 17 » Representation of The Notation

Suppose that $n = 2$, $X^+ = \{(-1, 5), (5, -1), (7/5, 7/5)\}$ *and* $X^- = \{(1, 1)\}$, *with the points marked in Figure 6.1(a). Then,* $\Lambda(X)$ *is* $\{(-1, 2), (2, -1), (1/5, 1/5)\}$, $I(X) = \{1, 2, 3\}$, *and* $G(X)$ *is the shaded region in Fig-*

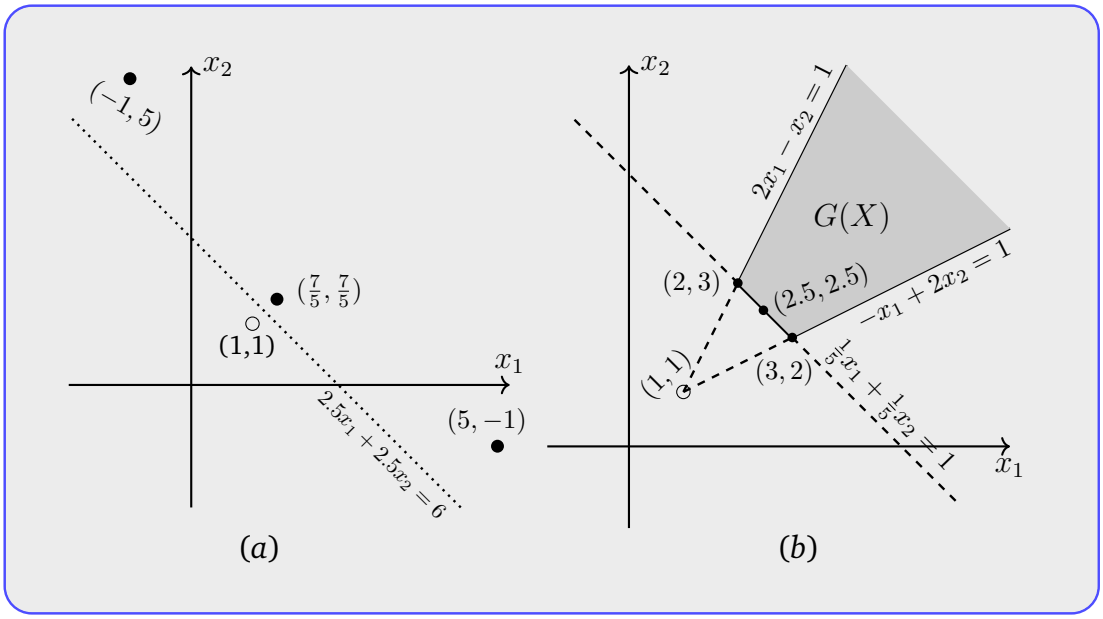

Figure 6.1: (a) The input samples discussed in Example 17 are shown as black (positive class) and white (negative class) circles. (b) The shaded region shows $G(X)$, with every element of the line segment between $(3,2)$ and $(2,3)$ being rescale-optimal in $G(X)$.

ure 6.1(b). For $w = (2.5, 2.5)$ (which is in $G(X)$), MP_w and MN_w are 7 and -5 respectively.

By assuming a linear relationship between the feature vector and the class label, as we showed in Section 2.3, each input point $(x, y) \in X$ expresses a linear constraint $y(w \cdot x + b) \geq 1$ with an unknown weight vector $w \in \mathbb{R}^n$ and unknown intercept term $b \in \mathbb{R}$. Thus, the feasible set $C(X)$ is defined as:

$$C(X) = \{(w, b) \in (\mathbb{R}^n, \mathbb{R}) : \forall (x, y) \in X, \quad y(w \cdot x + b) \geq 1\}.$$

To cope with the restriction caused by the linearity assumption of the model, we could transform x from $\mathbb{R}^n$ to a bigger space (say $\mathcal{H}$) by forming additional features based on the basic features. However, in this study, we assume this transformation has been explicitly defined; we do not consider making use of (non-linear) kernels [ABR64]. For certain problems, the linear kernel works sufficiently well, for instance, when the number of features is large [HCL+03, Appendix C].

As we discussed in Section 2.3, the main idea in SVM is that from the feasible set

$C(X)$ a pair (w, b) is chosen that maximises the margin; and this corresponds to the situation when w has the minimum norm in $\pi(C(X))$, where π is the projection function $\pi : (\mathbb{R}^n, \mathbb{R}) \to \mathbb{R}^n$ given by $\pi(w, b) = w$.

We will see in Proposition 43 below that $G(X) = \pi(C(X))$. This fact allows us to make use of general mathematical results from the previous chapter, which considers rescaling of features in preference learning. The following lemma is used to prove the proposition.

Lemma 42

Consider any finite and non-trivial $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ and $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$. Then the following all hold.

(i) $\min_{\lambda \in \Lambda(X)} w \cdot \lambda = \frac{1}{2}(MP_w + MN_w)$.

(ii) $(w, b) \in C(X)$ if and only if $1 - MP_w \leq b \leq MN_w - 1$.

(iii) $w \in G(X)$ if and only if $MP_w + MN_w \geq 2$, i.e., $1 - MP_w \leq MN_w - 1$. Also, if w has minimal norm in $G(X)$ then $MP_w + MN_w = 2$.

Proof: (i): $\min_{\lambda \in \Lambda(X)} w \cdot \lambda = \frac{1}{2} \min_{x^+ \in X^+, x^- \in X^-}(w \cdot x^+ - w \cdot x^-)$, which equals $\frac{1}{2}(\min_{x^+ \in X^+} w \cdot x^+ + \min_{x^- \in X^-} -w \cdot x^-)$, i.e., $\frac{1}{2}(MP_w + MN_w)$.

(ii): First assume that $(w, b) \in C(X)$. The definitions of $C(X)$ implies that for all $x^+ \in X^+$, $x^+ \cdot w + b \geq 1$, and for all $x^- \in X^-$, $-x^- \cdot w - b \geq 1$. This leads to $MP_w + b \geq 1$ and $MN_w - b \geq 1$, because $MP_w = \min_{x^+ \in X^+} w \cdot x^+$ and $MN_w = \min_{x^- \in X^-} w \cdot (-x^-)$. Rearranging gives $1 - MP_w \leq b \leq MN_w - 1$.

To prove the converse, suppose that $1 - MP_w \leq b \leq MN_w - 1$. Then, $MP_w + b \geq 1$ and $MN_w - b \geq 1$. The first inequality implies that for all $x^+ \in X^+$, $w \cdot x^+ + b \geq 1$, and the second inequality implies that for all $x^- \in X^-$, $-w \cdot x^- - b \geq 1$, and thus $-(w \cdot x^- + b) \geq 1$. Together these imply that $(w, b) \in C(X)$.

(iii): $w \in G(X)$ if and only if for all $\lambda \in \Lambda(X)$, $w \cdot \lambda \geq 1$, which is if and only if $\min_{\lambda \in \Lambda(X)} w \cdot \lambda \geq 1$. By part (i), this holds if and only if $MP_w + MN_w \geq 2$.

Table 6.1: The glossary of symbols being used throughout this chapter.

Symbol	Meaning
n	number of features.
X^+	$\subset \mathbb{R}^n$, input samples with positive labels; e.g., black circles in Figure 6.1(a).
X^-	$\subset \mathbb{R}^n$, input samples with negative labels; e.g., white circles in Figure 6.1(a).
X	$\subset \mathbb{R}^n \times \{+1, -1\}$, all input samples, i.e., the union of X^+ and X^- plus another dimension to include labels.
$\cdot$	the dot product, e.g., $(2,3) \cdot (3,1) = 9$.
$C(X)$	the feasible set defined as $\{(w,b) \in (\mathbb{R}^n, \mathbb{R}) \ : \ \forall (x,y) \in X, \quad y(w \cdot x + b) \geq 1\}$.
π	the projection function $\pi : (\mathbb{R}^n, \mathbb{R}) \to \mathbb{R}^n$ given by $\pi(w,b) = w$.
$\Lambda(X)$	defined as $\{\frac{x^+ - x^-}{2} \ : \ x^+ \in X^+, \ x^- \in X^-\}$.
$I(X)$	defined as $\{1, \ldots, \lvert X^+\rvert\lvert X^-\rvert\}$; i.e., the index set for $\Lambda(X)$.
$G(X)$	defined as $\{w \in \mathbb{R}^n \ : \ \forall \lambda \in \Lambda(X), \ w \cdot \lambda \geq 1\}$; e.g., the shaded region in Figure 6.1(b).
MP_w	for a given w is $\min_{x^+ \in X^+} w \cdot x^+$.
MN_w	for a given w is $\min_{x^- \in X^-} w \cdot (-x^-)$.
$\lVert w \rVert$	Euclidean norm of w.
w^*	the element with minimal norm in $G(X)$; e.g., $(2.5, 2.5)$ in Figure 6.1(b).
δ	$\in \mathbb{R}^n$, the translation (shift) vector.
τ	$\in \mathbb{R}^n_+$, the rescaling vector.
$\odot$	pointwise multiplication, e.g., $(2,3) \odot (3,1) = (6,3)$.
X_τ	defined as $\{(x \odot \tau, y) \ : \ (x,y) \in X\}$, i.e., X being rescaled by τ.
w^*_τ	the element with minimal norm in $G(X_\tau)$.
$\mathrm{RO}(X)$	defined as $\{w^*_\tau \odot \tau \ : \ \tau \in \mathbb{R}^n_+\}$; e.g., all elements on the line segment between $(3,2)$ and $(2,3)$ in Figure 6.1(b).

Assume that w has minimal norm in $G(X)$. Let $a_w = \min_{\lambda \in \Lambda(X)} w \cdot \lambda$. Since $w \in G(X)$, $a_w \geq 1$. Then $\frac{1}{a_w} w$ is an element of $G(X)$ with norm no more than the norm of w, and so has equal norm. This implies $a_w = 1$. By part (i), $MP_w + MN_w = 2$. ■

Proposition 43

Consider any finite and non-trivial $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ and $w \in \mathbb{R}^n$. Then, $w \in \pi(C(X))$ if and only if $w \in G(X)$. Thus, $\pi(C(X)) = G(X)$.

Proof: $w \in \pi(C(X))$ if and only if there exists $b \in \mathbb{R}$ such that $(w, b) \in C(X)$, which, by Lemma 42 part (ii) is if and only if there exists $b \in \mathbb{R}$ such that $1 - MP_w \leq b \leq MN_w - 1$. This holds if and only if $1 - MP_w \leq MN_w - 1$, i.e., $MP_w + MN_w \geq 2$, which, by Lemma 42 part (iii), holds if and only if $w \in G(X)$. ■

As is well-known, the solution that is picked by SVM from $C(X)$ is unique (see e.g., [BC99]). The following theorem restates this fact, making use of our notation.

Theorem 44: SVM Solution Uniqueness

Consider any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$. If $C(X)$ is non-empty then there exists a unique element $(w, b) \in C(X)$ such that w has minimal norm in $\pi(C(X))$. For that unique element, $b = 1 - MP_w = -1 + MN_w$.

Proof: By Proposition 43, $G(X) = \pi(C(X))$, which is non-empty because $C(X)$ is non-empty. $G(X)$ is a convex set, so there exists a unique element w in $G(X) = \pi(C(X))$ with minimal norm. Suppose that b is such

that $(w, b) \in C(X)$. Then, by Lemma 42(iii), $MP_w + MN_w = 2$, and thus, $1 - MP_w = MN_w - 1$. Lemma 42(ii) then implies that $b = 1 - MP_w = MN_w - 1$. Thus, there exists a unique b such that $(w, b) \in C(X)$. ■

In Example 17, $(2.5, 2.5)$ clearly is the unique element with minimal norm in $G(X)$, and the corresponding b is -6 ($= 1-MP = -1+MN$). Thus, its associated hyperplane $2.5x_1 + 2.5x_2 - 6 = 0$, the dotted line in Figure 6.1(a), produces the maximum margin.

Let us denote the solution of SVM, which by Theorem 44 is unique, by (w^*, b^*), where $b^* = 1 - MP_{w^*} = -1 + MN_{w^*}$. Thereafter, the feature vector $\alpha \in \mathbb{R}^n$ with unknown class label is classified as the positive ($+1$) class label if $w^* \cdot \alpha + b^* \geq 0$, and as the negative ($-1$) class label otherwise.

Theorem 44 leads easily to the following characterisation of classification, which is of a form that makes our extension in the following sections more straight-forward.

Proposition 45: Classification

Consider any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ and any $\alpha \in \mathbb{R}^n$. Then, the vector α is positively (resp. negatively) classified if and only if there exists $x^+ \in X^+$ (resp. $x^- \in X^-$) such that $w^* \cdot (x^+ - \alpha) \leq 1$ (resp. $w^* \cdot (\alpha - x^-) < 1$).

Proof: The vector α is positively (resp. negatively) classified if and only if $w^* \cdot \alpha + b^* \geq 0$ (resp. $w^* \cdot \alpha + b^* < 0$), which is, by Theorem 44, iff $w^* \cdot \alpha + 1 - MP_{w^*} \geq 0$ (resp. $w^* \cdot \alpha - 1 + MN_{w^*} < 0$), i.e., $MP_{w^*} \leq w^* \cdot \alpha + 1$ (resp. $MN_{w^*} < 1 - w^* \cdot \alpha$). This holds, from the definition of MP_{w^*} (resp. MN_{w^*}), if and only if $\min_{x^+ \in X^+} w^* \cdot x^+ \leq w^* \cdot \alpha + 1$ (resp. $\min_{x^- \in X^-} w^* \cdot (-x^-) < 1 - w^* \cdot \alpha$), which is if and only if there exists $x^+ \in X^+$ (resp. $x^- \in X^-$) such that $w^* \cdot x^+ \leq w^* \cdot \alpha + 1$ (resp. $-w^* \cdot x^- < 1 - w^* \cdot \alpha$). This immediately leads to the result. ■

From now on, we just work with the positive class; the results can be easily applied to the negative class as well.

6.3 Rescaling SVM

In this section we consider how performing certain affine transformations, i.e., translations and rescalings, on the domain of each feature, affect the result of SVM. We define a vector as being strongly positively classified, if it is positively classified under all affine transformations of each feature domain.

It is a known fact that the maximum margin hyperplane is essentially unaffected by a translation of feature space (see e.g., [Mei03]); i.e., by moving the origin, the normal vector to the separating hyperplane and hence the result of SVM does not change. Lemma 46 states this formally.

Lemma 46

Consider any finite and non-trivial $X \subseteq \mathbb{R}^n \times \{+1, -1\}$, any $\alpha, \delta \in \mathbb{R}^n$, and let $X^\delta = \{(x + \delta, y) : (x, y) \in X\}$. Then, α is positively classified with respect to X if and only if the vector $\alpha+\delta$ is positively classified with respect to X^δ.

Proof: $\Lambda(X^\delta)$ can be written as $\{\frac{(x^+ + \delta) - (x^- + \delta)}{2} : x^+ \in X^+, x^- \in X^-\}$, which is clearly equal to $\Lambda(X)$, and thus $G(X^\delta) = G(X)$. This implies that w^* for X and X^δ is the same.

Now, α is positively classified with respect to X, if and only if, by Proposition 45, there exists $x^+ \in X^+$ such that $w^* \cdot (x^+ - \alpha) \leq 1$, which is if and only if there exists $x^+ \in X^+$ such that $w^* \cdot ((x^+ + \delta) - (\alpha + \delta)) \leq 1$. This holds iff $\alpha + \delta$ is positively classified with respect to X^δ. ■

In contrast with translations, changing the scales of features may significantly affect the result of SVM. Consider the effect of a rescaling $\tau \in \mathbb{R}^n_+$, so that a feature vector $x \in \mathbb{R}^n$ is transformed into $x \odot \tau$ (recall the definition of $\odot$ from

Section 5.4). Therefore, X becomes $X_\tau = \{(x \odot \tau, y) : (x, y) \in X\}$. We write the element of $G(X_\tau)$ with minimal norm as w^*_τ.

Example 18 » The Effect of Rescaling on Classification

If we rescale X in Example 17 by $\tau = (5, 1)$ then X^+_τ will be $\{(-5, 5), (25, -1), (7, 7/5)\}$, X^-_τ becomes $\{(5, 1)\}$, and consequently, $\Lambda(X_\tau) = \{(-5, 2), (10, -1), (1, 1/5)\}$. It can be shown that w^*_τ equals $(3/5, 2)$ and $b^*_\tau = -6$. Now, let $\alpha = (-1, 4)$ and so $\alpha \odot \tau = (-5, 4)$. Clearly, when there is no scaling, α is positively classified since $(2.5, 2.5) \cdot (-1, 4) - 6 = 1.5 > 0$, but under scaling τ, $\alpha \odot \tau$ is negatively classified because $(3/5, 2) \cdot (-5, 4) - 6 = -1 < 0$.

For a rescaling vector $\tau \in \mathbb{R}^n_+$, we say that $\alpha \in \mathbb{R}^n$ is *positively classified under rescaling* τ iff $\alpha \odot \tau$ is positively classified with respect to X_τ, which, by using Proposition 45, is if and only if there exists $x^+_\tau \in X^+_\tau$ such that $w^*_\tau \cdot (x^+_\tau - (\alpha \odot \tau)) \leq 1$. This holds iff there exists $x^+ \in X^+$ such that $w^*_\tau \cdot ((x^+ \odot \tau) - (\alpha \odot \tau)) \leq 1$, i.e., $(w^*_\tau \odot \tau) \cdot (x^+ - \alpha) \leq 1$.

Let us say that $\alpha \in \mathbb{R}^n$ is *strongly* positively classified if and only if it is positively classified under any rescaling $\tau \in \mathbb{R}^n_+$ and any shift $\delta \in \mathbb{R}^n$. This is if and only if it is positively classified under any rescaling and no shift (i.e., a shift of $\delta = \mathbf{0}$, the zero vector).

Definition 39: Strongly Positive Classification

For $\alpha \in \mathbb{R}^n$, we define $Y_X(\alpha) = 1$ if and only if α is strongly positively classified; i.e., for all $\tau \in \mathbb{R}^n_+$ there exists $x^+ \in X^+$ such that $(w^*_\tau \odot \tau) \cdot (x^+ - \alpha) \leq 1$.

6.4 Characterisations Using Rescale Optimality

Here we use the notion of rescale-optimality from Section 5.4, and make use of general mathematical results from there. This leads to the computational

technique in Section 6.5 for testing if a vector is strongly positively classified. For the sake of convenience, we repeat the definition of rescale-optimal here with respect to the notation of this chapter.

Definition 40: rescale-optimal

For any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$, and $w \in G(X)$, we say that w is rescale-optimal in $G(X)$ if there exists $\tau \in \mathbb{R}^n_+$ such that for all $u \in G(X)$, $\|u \odot \tau\|^2 \geq \|w \odot \tau\|^2$.

It can be seen intuitively that every element of the line segment between $(3, 2)$ and $(2, 3)$ in Figure 6.1(b) is rescale-optimal; if $\tau(1) > \tau(2)$ (i.e., with the ratio $\frac{\tau(1)}{\tau(2)}$ being increased) then $w^*_\tau \odot \tau$ moves from $(2.5, 2.5)$ towards $(3, 2)$. Similarly, increasing the ratio $\frac{\tau(2)}{\tau(1)}$ from 1 moves $w^*_\tau \odot \tau$ from $(2.5, 2.5)$ towards $(2, 3)$. For instance, in Example 18, $\tau(1) = 5\tau(2)$ leads to $w^*_\tau \odot \tau = (3, 2)$.

We define $\mathrm{RO}(X)$ to be $\{w^*_\tau \odot \tau \,:\, \tau \in \mathbb{R}^n_+\}$ (in the previous chapter we used the term $\mathrm{SF}(\Lambda)$ for this set). Hence, it is clear, from Definition 39, that $Y_X(\alpha) = 1$ if and only if

$$\forall w \in \mathrm{RO}(X),\ \exists x^+ \in X^+,\ \text{s.t. } w \cdot (x^+ - \alpha) \leq 1.$$

The following proposition, which follows immediately from Proposition 20, states that the set of all rescale-optimal elements of $G(X)$ is $\mathrm{RO}(X)$.

Proposition 47

Consider any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$, and any $w \in \mathbb{R}^n$. Then, w is rescale-optimal in $G(X)$ if and only if w is in $\mathrm{RO}(X)$.

As a result of the equivalence of $\mathrm{RO}(X)$ to the set of all rescale-optimal elements of $G(X)$, we have the following proposition.

Proposition 48

Consider any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$, and any $\alpha \in \mathbb{R}^n$. Then, $Y_X(\alpha) = 1$ if and only if for every rescale-optimal element w in $G(X)$, there exists $x^+ \in X^+$ such that $w \cdot (x^+ - \alpha) \leq 1$.

Theorem 38 in Section 5.4 expresses rescale-optimal elements in terms of positive linear combinations; this theorem is used in Theorem 49 below which leads to a computational procedure for strong classification.

Theorem 49

Consider any finite non-trivial $X \subseteq \mathbb{R}^n \times \{+1, -1\}$, and any non-zero vector $w \in G(X)$. Then, w is rescale-optimal in $G(X)$ if and only if there exists $\mu \in \mathbb{R}^n$ and non-negative reals r_i, for each $i \in I(X)$, such that (a) μ agrees on signs with w; (b) $\mu = \sum_{i \in I} r_i \lambda_i$; and (c), for each $i \in I(X)$ either $r_i = 0$ or $\lambda_i \cdot w = 1$, where λ_i is the i^{th} element of $\Lambda(X)$.

Proof: Theorem 38 implies that w is rescale-optimal in $G(X)$ if and only if there exists $\mu \in \mathbb{R}^n$ and non-negative reals r_i (for each $i \in I(X)$) such that conditions (a), (b), (c) and (d) hold, where (a), (b) and (c) are the conditions above, and (d) is the condition that $w \cdot \mu = 1$.

$\Rightarrow$: this follows immediately from Theorem 38.

$\Leftarrow$: Assume that there exists $\mu \in \mathbb{R}^n$ and non-negative reals r_i (for each $i \in I(X)$) such that conditions (a), (b), (c) hold. w is not the zero vector, since $w \in G(X)$. Condition (a), that μ agrees on signs with w, implies that $\mu \cdot w > 0$. Let $\mu' = \frac{\mu}{\mu \cdot w}$, and for each $i \in I(X)$ define $r'_i = \frac{r_i}{\mu \cdot w}$. Then, (a) μ' agrees on signs with w, (b) $\mu' = \sum_{i \in I} r'_i \lambda_i$, (c) for each $i \in I(X)$ either $r'_i = 0$ or $\lambda_i \cdot w = 1$, and (d) $w \cdot \mu' = 1$. Theorem 38 then implies that w is rescale-optimal in $G(X)$. ■

In the following section, we characterise the situations in which rescaling the values of the features makes no difference to the result of the classification, in which case, strong classification is the same as the standard classification.

6.4.1 Determining Invariance to Rescaling

Theorem 50 below shows that rescaling the features vector makes no difference in the classification when there is a unique rescale-optimal vector in $G(X)$ (and the vector thus has the minimal norm in $G(X)$, using the identity rescaling). Recall that in the previous chapter, we already characterised when there is a unique rescale-optimal vector in Theorem 25, and proposed a polynomial algorithm to determine that unique element in Corollary 26.

Theorem 50

Consider any non-trivial finite $X \subseteq \mathbb{R}^n \times \{+1, -1\}$. Let *Pos*$(X)$ be the set of all $\alpha \in \mathbb{R}^n$ that are positively classified, and let *SPos*(X) be the set of α that are strongly positively classified. Then *Pos*$(X) =$ *SPos*(X) if and only if there exists a unique rescale-optimal element in $G(X)$, i.e., $\mathrm{RO}(X)$ is a singleton set.

Proof: For $w \in \mathbb{R}^n$, let *Pos*$_w$ be the set of all $\alpha \in \mathbb{R}^n$ such that there exists $x^+ \in X^+$ such that $w \cdot (x^+ - \alpha) \leq 1$. Then, by Proposition 45, *Pos*$(X) =$ *Pos*$_{w^*}$, and using Proposition 48, *SPos*(X) is the intersection of *Pos*$_w$ over all rescale-optimal w in $G(X)$. Note that w^* is rescale optimal in $G(X)$ (using the identity rescaling). Thus, if there is a single rescale-optimal w then $w = w^*$ and so *Pos*$(X) =$ *SPos*(X).

To prove the converse, suppose that *Pos*$(X) =$ *SPos*(X), which implies that for all rescale-optimal w in $G(X)$, *Pos*$_w$ contains *Pos*$_{w^*}$. Now, we can write *Pos*$_w$ as $\{\alpha \in \mathbb{R}^n : \alpha \cdot w \geq MP_w - 1\}$ because $MP_w = \min_{x^+ \in X^+} w \cdot x^+$, and thus *Pos*$_w$ is a half-space with normal vector w. Hence, since *Pos*$_w$ contains *Pos*$_{w^*}$, there exists a real scalar $q > 0$ with $w^* = qw$. Then for any $\tau \in \mathbb{R}^n_+$, $\|w^* \odot \tau\|^2 = q^2 \|w \odot \tau\|^2$. By Definition 40, $q = 1$, since both w and w^*

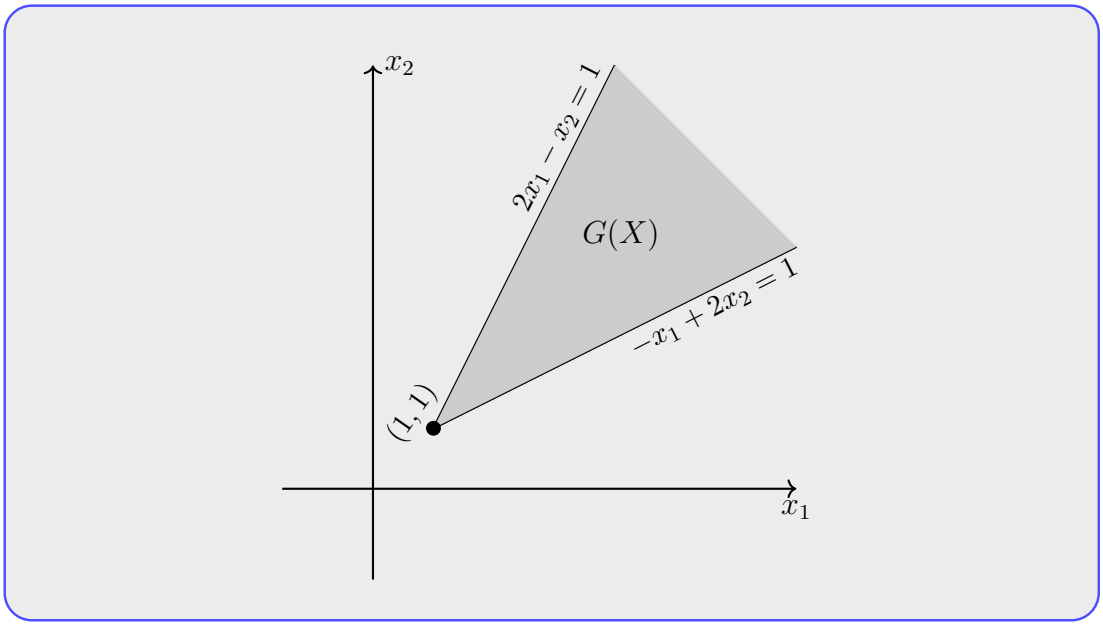

Figure 6.2: The shaded region shows $G(X)$ for the case explained in Example 19, with $(1,1)$ being the unique rescale-optimal element in $G(X)$.

are rescale-optimal, and hence $w = w^*$. This shows that there is a unique rescale-optimal element in $G(X)$. ■

Example 19 » Invariance to Rescaling

Consider modifying Example 17 by just removing $(7/5, 7/5)$ *from* X^+. *Then,* $\Lambda(X) = \{(-1,2),(2,-1)\}$, *and* $G(X)$ *becomes the intersection of the half-spaces* $-x_1 + 2x_2 \geq 1$ *and* $2x_1 - x_2 \geq 1$, *with a single extremal point* $(1,1)$. *This point pointwise dominates every other element in* $G(X)$ *(see Figure 6.2). Pointwise dominating every other element, from Theorem 25, means that* $(1,1)$ *will be the unique element of* $G(X)$ *that is rescale-optimal. Using Theorem 50, this implies that the classification is invariant to the rescaling (in this example).*

6.5 Computation of Strong Classification

Here we express if an instance is strongly positively classified in terms of a set of constraints.

For a set $X \subseteq \mathbb{R}^n \times \{+1, -1\}$ and arbitrary $\alpha \in \mathbb{R}^n$, we would like to determine if $Y_X(\alpha) = 1$, i.e., if α is strongly positively classified. We can infer from Proposition 48 that $Y_X(\alpha) \neq 1$ if and only if there exists a rescale-optimal element w in $G(X)$ such that for all $x^+ \in X^+$, $w \cdot (x^+ - \alpha) > 1$. By taking into account Theorem 49, we obtain a set of inequalities to determine if $Y_X(\alpha) \neq 1$ as follows.

Let λ_i be the i^{th} element of $\Lambda(X)$ where $i \in I(X)$. Now, $Y_X(\alpha) \neq 1$ if and only if there exists $w \in \mathbb{R}^n$ and $\mu \in \mathbb{R}^n$, and non-negative reals r_i for each $i \in I(X)$ such that

- $\forall x^+ \in X^+$, $w \cdot (x^+ - \alpha) > 1$;
- $\forall i \in I(X)$, $w \cdot \lambda_i \geq 1$; (i.e., $w \in G(X)$)
- $\forall i \in I(X)$, [$w \cdot \lambda_i = 1$ or $r_i = 0$];
- $\forall j = 1, \ldots, n$, $w(j) = 0 \iff \mu(j) = 0$, and $w(j) > 0 \iff \mu(j) > 0$; (i.e., agreeing on signs);
- $\mu = \sum_{i \in I(X)} r_i \lambda_i$.

The number of constraints here is $2|X^+||X^-| + |X^+| + n + 2$.

6.6 Experimental Results

The experiments make use of the *UCI machine learning repository*[1] from which six real data sets are chosen, namely *Breast Cancer Wisconsin* [SWM93], *Pima Indians Diabetes*, *Blood Transfusion Service Center* [YYT09], *Indian Liver Patient*, *Fertility* [MGDJ+12], and *Banknote Authentication*. In fact, we include data sets that meet three criteria: (i) the data set has only two classes, (ii) the data set consists of only numeric features, and (iii) number of features is at most 10. The first two criteria ensure that the data set complies with the proposed method in this chapter, and the third made the computation especially fast. Thereafter, 18

[1] http://archive.ics.uci.edu/ml/

Table 6.2: The specifications of 18 benchmarks which are used for the experiments; these are derived from six real data sets.

Bench.	$\|X\|$	$\|I(X)\|$	Data Set	Features (n)
1.	16	63		
2.	20	84	Breast Cancer Wisconsin	9
3.	25	126		
4.	22	72		
5.	28	171	Pima Indians Diabetes	8
6.	18	65		
7.	20	64		
8.	25	46	Blood Transfusion	4
9.	25	144		
10.	25	84		
11.	15	36	Indian Liver Patient	10
12.	21	108		
13.	30	104		
14.	20	81	Fertility	9
15.	15	14		
16.	25	156		
17.	20	100	Banknote Authentication	4
18.	15	50		

benchmarks are derived from those data sets. Table 6.2 contains specifications of these benchmarks.

For constructing a benchmark, a random selector creates two disjoint sets from a data set, one for learning (i.e., X) and one for testing. However, a preprocessing phase, similar to the first procedure described in Section5.6, deletes some elements of the learning set in order to make it consistent (i.e., $C(X) \neq \emptyset$), since we consider only the consistent case.

For each instance in the testing set (say α), the rescaling method determines—based on the learning set X—either (i) it is strongly positive ($Y_X(\alpha) = 1$), or (ii) it is strongly negative ($Y_X(\alpha) = -1$), or (iii) it is neutral ($Y_X(\alpha) \neq 1$ and $Y_X(\alpha) \neq -1$).

The number and percentage of neutral instances for each benchmark can be found in Table 6.3. The ratio of neutral instances for the benchmarks (i.e., those instances being classified differently under different rescaling) varies from 6% to 95% with a mean of 55.9%. That means that for more than half of the instances, rescaling the feature values can change the result of the SVM classifi-

cation. That points out the unreliability of standard SVM–specifically for larger numbers like 95%–with respect to being sensitive to the way the features are scaled.

In a testing set, among the non-neutral instances, we can also count:

1. The number of negative instances improperly *strongly* classified as positive (False Positive).
2. The number of positive instances properly *strongly* classified as positive (True Positive).
3. The number of positive instances improperly *strongly* classified as negative (False Negative).
4. The number of negative instances properly *strongly* classified as negative (True Negative).

Conventional SVM can also predict a class label for each instance, positive or negative. As a result, we have False Positive (FP), True Positive (TP), False Negative (FN) and True Negative (TN) for SVM as well. Tables 6.4 and 6.5 compare these measures between the two methods. Note that the value of FP + TN (resp. FN + TP) for the rescaling method is not (necessarily) equal to the total number of the testing instances with real negative (resp. positive) class label because some instances are classified as neutral.

In Tables 6.4 and 6.5, we also compare the *Positive Predictive Value* (PPV) and *Negative Predictive Value* (NPV) of the two methods. PPV is the fraction of truly positive instances among positively classified ones. Similarly, NPV is the fraction of truly negative instances among negatively classified ones. Thus,

$$\text{PPV} = \frac{\text{TP}}{\text{FP} + \text{TP}}, \text{ and}$$
$$\text{NPV} = \frac{\text{TN}}{\text{FN} + \text{TN}}.$$

The measures PPV and NPV are widely used in medical contexts in order to validate a disease diagnostic test [PMP+08]. To illustrate, PPV expresses that if a patient's test is positive how likely it is that he really has the disease. Similarly, NPV means that if a patient's test is negative how likely it is that she really does not have the disease. The results show an increase of 5.5% in PPV and NPV for the benchmarks on average.

Note that our approach is not intended to be a competitor of SVM in terms

Table 6.3: The number and ratio of instances labelled as neutral by the rescaling method are shown for each benchmark.

Benchmark	Neutral #	Total #	Ratio(%)
1.	261	683	38
2.	53	100	53
3.	53	100	53
4.	186	200	93
5.	168	200	84
6.	91	100	91
7.	35	100	35
8.	6	100	6
9.	9	100	9
10.	72	100	72
11.	93	100	93
12.	191	200	95
13.	46	70	66
14.	56	81	69
15.	79	85	93
16.	14	100	14
17.	12	100	12
18.	30	100	30
Mean			55.9

of classification accuracy. What it does is to highlight certain instances where we can have greater confidence, with the other instances having reduced confidence because the result of the classification could be changed if a different scaling of the features were used. This is why PPV and NPV are appropriate measures for the experimental results.

The approach discussed in Section 6.5 was implemented using the solver CPLEX 12.6.2. Determining whether an instance is strongly positive or strongly negative or neutral for any of benchmarks takes less than a couple of seconds, making use of a computer facilitated by a Core i7 2.60 GHz processor and 8 GB RAM memory.

6.7 Discussion and Summary

The scaling of individual features, before applying an SVM method, can be subjective and arbitrary. However, the way features are scaled can make a difference; in fact, for a little more than half of the instances in our experiments,

Table 6.4: A comparison, using 18 benchmarks, between the PPV of SVM and the rescaling method. The undefined values, labelled -, are excluded from the mean.

Bench.	SVM			Rescaling Method		
	FP	TP	PPV(%)	FP	TP	PPV(%)
1.	15	439	97	0	333	**100**
2.	3	71	96	0	46	**100**
3.	9	64	88	0	47	**100**
4.	25	48	**66**	1	1	50
5.	38	53	58	1	2	**67**
6.	8	15	65	0	0	-
7.	25	6	19	5	4	**44**
8.	7	4	**43**	4	3	36
9.	27	9	**25**	25	8	24
10.	15	13	46	0	2	**100**
11.	35	15	**30**	1	0	0
12.	44	23	34	3	4	**57**
13.	15	2	**12**	1	0	0
14.	19	1	5	3	1	**25**
15.	11	3	21	0	0	-
16.	3	45	**94**	3	38	93
17.	1	42	**98**	1	38	97
18.	3	44	94	0	32	**100**
Mean			56.56			**62.06**

rescaling the feature values can change the result of the SVM classification. Based on this fact, we say an instance strongly belongs to a class if it belongs to that class for all rescalings of features. This new definition boosts the confidence of labelling instances by excluding those instances which are classified differently under different scalings.

Building on the general mathematical results of the previous chapter, we have developed a computational procedure that can test if an instance is strongly positive, i.e., labelled positive for every affine way of rescaling the values of each feature (and similarly for the negative case). We also have a polynomial algorithm for determining when, for a given training set, the classification of any possible instance is not affected by rescaling. Our experiments also showed a slight improvement in the value of prediction, i.e., the likelihood that if an instance is classified as a particular class, it truly belongs to that class.

Table 6.5: A comparison, using 18 benchmarks, between the NPV of SVM and the rescaling method. The undefined values, labelled -, are excluded from the mean.

Bench.	SVM			Rescaling Method		
	FN	TN	NPV(%)	FN	TN	NPV(%)
1.	10	219	96	3	86	**97**
2.	1	25	96	0	1	**100**
3.	0	27	100	0	0	-
4.	22	105	83	0	12	**100**
5.	14	95	87	0	29	**100**
6.	21	56	73	0	9	**100**
7.	17	52	75	11	45	**80**
8.	18	71	80	18	69	**79**
9.	9	55	**86**	8	50	**86**
10.	19	53	**74**	9	17	65
11.	13	37	**74**	2	4	67
12.	27	106	80	3	4	**100**
13.	6	47	89	1	22	**96**
14.	6	55	90	1	20	**95**
15.	8	63	89	0	6	**100**
16.	2	50	96	0	45	**100**
17.	1	56	98	0	49	**100**
18.	2	51	96	0	38	**100**
Mean			86.00			**91.47**

There are many potential future directions following from this work; for instance, extending the method to cope with inconsistencies in a more sophisticated way (i.e., soft margin SVM); attempting to develop the computational method for certain kernels; performing multi-class classification by making use of repeated binary-class classification (e.g., using one-vs-all or one-vs-one approaches); and computing variations of the method where the user can limit the rescaling range for different features.

Chapter 7

Conclusion

In this chapter, we discuss what has been achieved in the thesis, and we describe some possible future directions.

7.1 Summary

There is an increasing need for effective and reliable techniques to assist users in decision-making problems; e.g., choosing an item from a set of alternatives. Examples of these items could be products, movies, vacation packages or cameras. Any approach that is adopted to fulfil this demand should be able to take into account user preferences. In this thesis, we have made some contributions towards user preferences learning especially based on the maximum margin method.

In Chapter 4, we applied the maximum margin preference relations (referred to as SVPL) to a ridesharing application. Ridesharing can potentially have many benefits such as reducing traffic congestion, pollution and the cost of travel, but matching drivers and riders as optimal as possible is a main challenge. In contrast to the traditional ride-matching systems, we have presented a matching approach in which the user's preferences contribute to the matching process. This approach learns user preferences from past ridesharing records of the user and incorporate them to recommend better matchings. Our experiments have shown that considering the user preferences can improve the ranking accuracy of ride opportunities, and thus users' satisfaction degrees.

However, as we observed in Chapter 5, the maximum margin preference relation, $\succcurlyeq^{mm}$, is sensitive to the rescaling of preference inputs or features. However, it makes sense for a preference relation to be scaling invariant; regarding the invariance to the rescaling of preference inputs, if the user prefers the feature vector a to b, then it is often expected that $2a$ is also preferred to $2b$. With respect to the invariance to the invariance to the rescaling of features, while scaling of features is a crucial pre-processing task, the way that it is done it can highly depend on the received instances. As a result, this idea led us to develop three preference relations that are more robust in a sense that they are invariant to the scaling of (i) preference inputs ($\succcurlyeq^{\mathrm{I}}_{\Lambda}$), (ii) features ($\succcurlyeq^{\mathrm{F}}_{\Lambda}$), and (iii) preference inputs and features simultaneously ($\succcurlyeq^{\mathrm{I,F}}_{\Lambda}$). We investigated the relationships between these preference relations, along with two other relations $\succcurlyeq^{mm}$ and $\succcurlyeq^{\mathrm{C}}_{\Lambda}$. Also, we presented two incremental algorithms to find the optimal/best

elements with respect to these relations; (i) the set of undominated elements consisting of the elements that there is no other element that can strictly dominate them, and (ii) the set of possibly optimal elements which consists of the elements that are optimal in some scenario (i.e., some scaling). The experiments, which were carried out on two real databases, highlighted the level of decisiveness, the computation speed, and the number of the optimal elements for each preference relation.

Rescaling of the domain of each feature is not only an issue for the maximum margin preference relation, but also it can change the result of the standard SVM in the classification task. Because of this we proposed, in Chapter 6, a kind of classification method that is invariant to the way that features are scaled. One merit of this method is that it identifies those instances that are classified differently under some different scaling. The experiments showed an increase in the likelihood that if an instance is classified as a particular class, it truly belongs to that class.

7.2 Possible Future Work

There are several future directions which will be worth investigating; in this section, we briefly discuss some of those.

More General Preference Statements: In this thesis, we considered that preferences inputs are stated in the form of pairwise comparisons; i.e., the feature vector $a_i \in \mathbb{R}^n$ has been preferred to $b_i \in \mathbb{R}^n$, for each $i \in \{1, \ldots, m\}$. One extension would be to also take into account more general kinds of input preference statements such as a *ceteris paribus* semantics (as being used in a CP-net). For example, in the ridesharing application, Alice may state that "I prefer a female rider to share my trip with, provided all other features are the same".

Uncertainty in Preference Statements: It would be also interesting to extend our methods to consider a level of confidence (or certainty) associated with each preference statement. That could naturally arise in many decision-making problems especially when we have implicit preference elicitation. For instance, in a situation where there is no background information about a new user, we could make use of users' preferences (votes) and might, for instance, imply that the new user will prefer the

hotel A to B with probability of 65%. There are some approaches in the classification context that could be helpful here, in which each training instance is associated with a weight (see e.g., [YSW07]).

Using The Kernel Trick: As we have seen, our rescaling-invariant methods for preference learning and classification are built on an assumption that the (utility) function is linear. However, as we explained in Section 4.4.3, the kernel trick can be used to manage non-linearly-representable data in a linear manner. Thus, one interesting direction for future work would be to find computational methods for certain kernels. To do so, we need to rewrite the computation method such that (i) all unknown variables that live in $\mathbb{R}^n$ vanish, and (ii) all alternatives appear in the form of dot products. We can do this in a straight-forward way for the $\succcurlyeq^{\mathrm{I}}_{\Lambda}$ relation as follows. Suppose that the feature vectors are mapped from $\mathbb{R}^n$ to a higher dimensional space $\mathcal{H}$ with the function Φ. Therefore, by seeing Section 5.7.2, $\alpha \not\succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ if and only if there exists $\Phi(u) \in \mathcal{H}$, and non-negative reals r_i for each $i \in I$, such that

- $\Phi(u) \cdot (\Phi(\beta) - \Phi(\alpha)) > 0$;
- $\forall i \in I$, $\Phi(u) \cdot \Phi(\lambda_i) \geq 1$; and
- $\Phi(u) = \sum_{i \in I} r_i \Phi(\lambda_i)$.

By using the third constraint, we can eliminate $\Phi(u)$ which is an unknown variable. Hence, $\alpha \not\succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ if and only if there exists non-negative reals r_i for each $i \in I$, such that

- $(\sum_{i \in I} r_i \Phi(\lambda_i)) \cdot (\Phi(\beta) - \Phi(\alpha)) > 0$;
- $\forall i \in I$, $(\sum_{j \in I} r_j \Phi(\lambda_j)) \cdot \Phi(\lambda_i) \geq 1$.

Now, we consider the kernel function $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ such that $K(x, y) = \Phi(x) \cdot \Phi(y)$. So, we have, $\alpha \not\succcurlyeq^{\mathrm{I}}_{\Lambda} \beta$ if and only if there exists non-negative reals r_i for each $i \in I$, such that

- $\sum_{i \in I} r_i (K(\lambda_i, \beta) - K(\lambda_i, \alpha)) > 0$;
- $\forall i \in I$, $\sum_{j \in I} r_j K(\lambda_j, \lambda_i) \geq 1$.

Data Inconsistency Handling: We described some approaches in Section 5.6 in order to restore the consistency in the data. Nonetheless, a more sophisticated method, in which no pre-processing on the data is needed, might be more desirable. That could, for example, include introducing a

set of parameters representing the cost of violating each constraint, and changing the computation methods to minimise the total cost.

Restricting The Rescaling Range: In the features-rescaling-invariant methods (either in the preference learning context or the classification), it may seem natural to explore variations of the method where the user can restrict the rescaling range for different features. i.e., for $j \in \{1, \ldots, n\}, \tau(j) \in (0, UB_j]$ with $UB_j \in \mathbb{R}_+$ (instead of $\tau(j) \in \mathbb{R}_+$).

Multi-Class Classification: One obvious expansion of the rescaling-invariant classification method is to perform rescaling-invariant multi-class classification. That means if we have three classes $c_1, c_2,$ and c_3, an instance is strongly classified as c_1 if it is classified as c_1 for all the choices of scaling of features. One way to do that would be to adopt some well-known strategies such as one-vs-all and one-vs-one [Bis06] to repeat the binary classification. For example, in one-vs-all strategy, the training samples of classes c_2 and c_3 are grouped into one, and then we can run the binary rescaling-invariant method to find those instances that strongly belong to the c_1 class. In the same way, the classes c_1 and c_3 can make a group, and then strongly classified c_2 instances will be determined.

References

[AAM11] Andrew Amey, John Attanucci, and Rabi Mishalani. Real-time ridesharing – the opportunities and challenges of utilizing mobile phone technology to improve rideshare services. *Transportation Research Record: Journal of the Transportation Research Board*, 2217(1):103–110, 2011.

[ABR64] MA Aiserman, EM Braverman, and LI Rozonoer. Theoretical foundations of the potential function method in pattern recognition. *Avtomat. i Telemekh*, 25:917–936, 1964.

[AESW11] Niels AH Agatz, Alan L Erera, Martin WP Savelsbergh, and Xing Wang. Dynamic ride-sharing: A simulation study in metro Atlanta. *Transportation Research Part B: Methodological*, 45(9):1450–1464, 2011.

[AESW12] Niels Agatz, Alan Erera, Martin Savelsbergh, and Xing Wang. Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2):295 – 303, 2012.

[AH01] Selim Aksoy and Robert M. Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recognition Letters*, 22(5):563–582, 2001.

[ASBP13] Ehsan Abbasnejad, Scott Sanner, Edwin V. Bonilla, and Pascal Poupart. Learning community-based preferences via Dirichlet process mixtures of Gaussian processes. In *Proceedings of The 23rd International Joint Conference on Artificial Intelligence, IJCAI 2013, Beijing, China, August 3-9, 2013*, pages 1213–1219. IJCAI/AAAI, 2013.

[Atk70] Anthony B Atkinson. On the measurement of inequality. *Journal of Economic Theory*, 2(3):244–263, 1970.

[BB05] Darius Braziunas and Craig Boutilier. Local utility elicitation in GAI models. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence, UAI 2005*, pages 42–49. AUAI Press, 2005.

[BB07] Darius Braziunas and Craig Boutilier. Minimax regret based elicitation of generalized additive utilities. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2007*, pages 25–32, 2007.

[BB12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

[BBB01] Craig Boutilier, Fahiem Bacchus, and Ronen I Brafman. UCP-networks: A directed graphical representation of conditional utilities. In *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence, UAI 2001*, pages 56–64. Morgan Kaufmann Publishers Inc., 2001.

[BBD+04] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Poole. CP-nets: A tool for reasoning with conditional *ceteris paribus* preference statements. *Journal of Artificial Intelligence Research*, 21(1):135–191, February 2004.

[BBHP99] Craig Boutilier, Ronen I Brafman, Holger H Hoos, and David Poole. Reasoning with conditional *ceteris paribus* preference statements. In *Proceedings of the Fifteenth conference on Uncertainty in Artificial Intelligence, UAI 1999*, pages 71–80. Morgan Kaufmann Publishers Inc., 1999.

[BC99] Christopher JC Burges and David J Crisp. Uniqueness of the SVM solution. In *Advances in Neural Information Processing Systems 12, NIPS 1999*, volume 99, pages 223–229, 1999.

[BDS06] Ronen I Brafman, Carmel Domshlak, and Solomon Eyal Shimony. On graphical modeling of preference and importance. *Journal of Artificial Intelligence Research*, 25:389–424, 2006.

[Ber05] Dennis S Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas with Application to Linear Systems Theory*, volume 41. Princeton University Press, 2005.

[BG95] Fahiem Bacchus and Adam Grove. Graphical models for preference and utility. In *Proceedings of the Eleventh conference on Uncertainty in Artificial Intelligence, UAI 1995*, pages 3–10. Morgan Kaufmann Publishers Inc., 1995.

[BGH10] Adriana Birlutiu, Perry Groot, and Tom Heskes. Multi-task preference learning with an application to hearing aid personalization. *Neurocomputing*, 73(7):1177–1185, 2010.

[BGW09] Radu I. Bot, Sorin-Mihai Grad, and Gert Wanka. *Duality in Vector Optimization*. Springer, 2009.

[BHK98] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in Artificial Intelligence, UAI 1998*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[BHW10] Asa Ben-Hur and Jason Weston. *A User's Guide to Support Vector Machines*, pages 223–239. Humana Press, Totowa, NJ, 2010.

[BHY96] Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. Knowledge-based navigation of complex information spaces. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence, AAAI 1996*, pages 462–468. AAAI Press, 1996.

[BHY97] Robin D Burke, Kristian J Hammond, and BC Yound. The FindMe approach to assisted browsing. *IEEE Expert*, 12(4):32–40, Jul 1997.

[Bis06] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

[BJ88] Salvador Barbarà and Matthew Jackson. Maximin, leximin, and the protective criterion: characterizations and comparisons. *Journal of Economic Theory*, 46(1):34–44, 1988.

[BKU02] Vladimir Bugera, Hiroshi Konno, and Stanislav Uryasev. Credit cards scoring with quadratic utility functions. *Journal of Multi-Criteria Decision Analysis*, 11(4-5):197–211, 2002.

[BMM04] Roberto Baldacci, Vittorio Maniezzo, and Aristide Mingozzi. An exact method for the car pooling problem based on Lagrangean column generation. *Operations Research*, 52(3):422–439, 2004.

[BMM08] Lamia Berrah, Gilles Mauris, and Jacky Montmain. Monitoring the improvement of an overall industrial performance based on a Choquet integral aggregation. *Omega*, 36(3):340–351, 2008.

[Bos02] Dustin Boswell. Introduction to support vector machines. `http://dustwell.com/PastWork/IntroToSVM.pdf`, 2002. [Online; accessed 9-May-2018].

[Bot91] Léon Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8), 1991.

[BP13] Géraldine Bous and Marc Pirlot. Learning multicriteria utility functions with random utility models. In *Proceedings of the Third International Conference on Algorithmic Decision Theory, ADT 2013, Bruxelles, Belgium, November 12-14*, pages 101–115. Springer Berlin Heidelberg, 2013.

[BP15] Nawal Benabbou and Patrice Perny. On possibly optimal tradeoffs in multicriteria spanning tree problems. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory, ADT 2015*, pages 322–337, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

[BPPS03] Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Constraint-based optimization with the minimax decision criterion. In *Principles and Practice of Constraint Programming, CP 2003*, pages 168–182. Springer, 2003.

[BPPS05] Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Regret-based utility elicitation in constraint-based decision problems. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, IJCAI 2005, Edinburgh, Scotland*, volume 9, pages 929–934, 2005.

[BPV17] Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence*, 246:152–180, 2017.

[BS01] Michel Beuthe and Giuseppe Scannella. Comparative analysis of UTA multicriteria methods. *European Journal of Operational Research*, 130(2):246–262, 2001.

[BS02] Evgeny Byvatov and Gisbert Schneider. Support vector machine applications in bioinformatics. *Applied Bioinformatics*, 2(2):67–77, 2002.

[BSR+05] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning, ICML 2005*, pages 89–96. ACM, 2005.

[Bur98] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[CBM96] Rich Caruana, Shumeet Baluja, and Tom Mitchell. Using the future to sort out the present: Rankprop and multitask learning for medical risk evaluation. In *Advances in Neural Information Processing Systems, NIPS 1996*, pages 959–965, 1996.

[CdLHM04] Roberto Wolfler Calvo, Fabio de Luigi, Palle Haastrup, and Vittorio Maniezzo. A distributed geographic information system for the daily car pooling problem. *Computers & Operations Research*, 31(13):2263–2278, 2004.

[CH15] Samprit Chatterjee and Ali S Hadi. *Regression Analysis by Example*. John Wiley & Sons, 2015.

[Cho54] Gustave Choquet. Theory of capacities. In *Annales de l'institut Fourier*, volume 5, pages 131–295, 1954.

[CKPQ10] Vineeta Chaube, Andrea L Kavanaugh, and Manuel A Perez-Quinones. Leveraging social networks to embed trust in rideshare programs. In *Proceedings of the 43rd Hawaii International Conference on System Sciences, HICSS 2010*, pages 1–8. IEEE, 2010.

[CS99] Andrew M Colman and Jonathan A Stirk. Singleton bias and lexicographic preferences among equally valued alternatives. *Journal of Economic Behavior & Organization*, 40(4):337–351, 1999.

[CS12] Nelson D Chan and Susan A Shaheen. Ridesharing in North America: Past, present, and future. *Transport Reviews*, 32(1):93–112, 2012.

[CSS99] William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10(1):243–270, May 1999.

[CT03] Li-Juan Cao and Francis Eng Hock Tay. Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on Neural Networks*, 14(6):1506–1518, 2003.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[Daw79] Robyn M Dawes. The robust beauty of improper linear models in decision making. *American Psychologist*, 34(7):571, 1979.

[DBK+97] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems, NIPS 1997*, pages 155–161, 1997.

[DD11] Anulekha Dhara and Joydeep Dutta. *Optimality Conditions in Convex Cptimization: a Finite-dimensional View*. CRC Press, 2011.

[DF05] Didier Dubois and Hélene Fargier. Bridging the gap between discrete Sugeno and Choquet integrals. In *Présentéa la conférence RUD*. Citeseer, 2005.

[DGP09] Jean-Philippe Dubus, Christophe Gonzales, and Patrice Perny. Fast recommendations using GAI models. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence, IJCAI 2009*, volume 9, pages 1896–1901, 2009.

[DHKP11] Carmel Domshlak, Eyke Hüllermeier, Souhila Kaci, and Henri Prade. Preferences in AI: An overview. *Artificial Intelligence*, 175(7-8):1037–1052, 2011.

[DPR+06] Carmel Domshlak, Steve Prestwich, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Hard and soft constraints for reasoning about qualitative conditional preferences. *Journal of Heuristics*, 12(4):263–285, 2006.

[DPT13] Didier Dubois, Henri Prade, and Fayçal Touazi. Conditional preference nets and possibilistic logic. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 181–193. Springer, 2013.

[DRVW09] Carmel Domshlak, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Reasoning about soft constraints and conditional preferences: complexity results and approximation techniques. *arXiv preprint arXiv:0905.3766*, 2009.

[DS03] Gyuri Dorkó and Cordelia Schmid. Selection of scale-invariant parts for object class recognition. In *Proceedings of the 9th IEEE International Conference on Computer Vision, ICCV 2003, 14-17 October 2003, Nice, France*, volume 1, pages 634–640, 2003.

[DS13] Maeve Duggan and Aaron Smith. Cell internet use 2013. Pew Research Center, http://www.pewinternet.org/2013/09/16/cell-internet-use-2013/, 2013.

[Dye05] James S Dyer. MAUT—multiattribute utility theory. In *Multiple Criteria Decision Analysis: State of The Art Surveys*, pages 265–292. Springer, 2005.

[DYZ90] Dimitris K. Despotis, Denis Yannacopoulos, and Constantin Zopounidis. A review of the UTA multicriteria method and some improvements. *Foundations of Computing and Decision Sciences*, 15(2):63–76, 1990.

[DZ95] Dimitris K. Despotis and Constantin Zopounidis. Building additive utilities in the presence of non-monotonic preferences. In *Advances in Multicriteria Analysis*, pages 101–114. Springer, 1995.

[Eck65] Robert T Eckenrode. Weighting multiple criteria. *Management science*, 12(3):180–192, 1965.

[Ehr99] Matthias Ehrgott. A characterization of lexicographic max-ordering solutions. `http://nbn-resolving.de/urn/resolver.`

pl?urn:nbn:de:hbz:386-kluedo-4531, 1999. [Online; accessed 9-May-2018].

[Ema14] Emarketer. 2 billion consumers worldwide to get smart(phones) by 2016. http://www.emarketer.com/Article/2-Billion-Consumers-Worldwide-Smartphones-by-2016/1011694, 2014.

[Far93] Hélene Fargier. Selecting preferred solutions in fuzzy constraint satisfaction problems. In *Proceedings of the 1st European Congress on Fuzzy and Intelligent Technologies, EUFIT 1993, Aachen, Germany, Sept. 7-10*, pages 1128–1134, 1993.

[FDO+13] Masabumi Furuhata, Maged Dessouky, Fernando Ordóñez, Marc-Etienne Brunet, Xiaoqing Wang, and Sven Koenig. Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57:28–46, 2013.

[FGS09] José Rui Figueira, Salvatore Greco, and Roman Słowiński. Building a set of additive value functions representing a reference preorder and intensities of preference: GRIP method. *European Journal of Operational Research*, 195(2):460–486, 2009.

[FH10] Johannes Fürnkranz and Eyke Hüllermeier. *Preference Learning*. Springer, 2010.

[FHWW10] Eugene C Freuder, Robert Heffernan, Richard J Wallace, and Nic Wilson. Lexicographically-ordered constraint satisfaction problems. *Constraints*, 15(1):1–28, 2010.

[Fis67] Peter C Fishburn. Interdependence and additivity in multivariate, unidimensional expected utility theory. *International Economic Review*, 8(3):335–342, 1967.

[Fis70] Peter C. Fishburn. *Utility Theory for Decision Making*. Wiley, 1970.

[Fis74] Peter C. Fishburn. Lexicographic orders, utilities and decision rules: A survey. *Management Science*, 20(11):1442–1471, 1974.

[FISS03] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *The Journal of Machine Learning Research*, 4:933–969, 2003.

[FRW10] Hélène Fargier, Emma Rollon, and Nic Wilson. Enabling local computation for partially ordered preferences. *Constraints*, 15(4):516–539, Oct 2010.

[FS95] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37. Springer, 1995.

[FSS+89] J Kevin Ford, Neal Schmitt, Susan L Schechtman, Brian M Hults, and Mary L Doherty. Process tracing methods: Contributions, problems, and neglected research questions. *Organizational Behavior and Human Decision Processes*, 43(1):75–117, 1989.

[Gal08] Jean Gallier. Notes on convex sets, polytopes, polyhedra, combinatorial topology, Voronoi diagrams and Delaunay triangulations. *arXiv preprint arXiv:0805.0292*, 2008.

[GH05] Mithat Gönen and Glenn Heller. Concordance probability and discriminatory power in proportional hazards regression. *Biometrika*, 92(4):965–970, 2005.

[GHH11] Keivan Ghoseiri, Ali Ebadollahzadeh Haghani, and Masoud Hamedi. *Real-time rideshare matching problem*. Mid-Atlantic Universities Transportation Center Berkeley, 2011.

[GKM08] Michel Grabisch, Ivan Kojadinovic, and Patrick Meyer. A review of methods for capacity identification in Choquet integral based multi-attribute utility theory: Applications of the Kappalab R package. *European Journal of Operational Research*, 186(2):766–785, 2008.

[GL05] Michel Grabisch and Christophe Labreuche. Bi-capacities—ii: the Choquet integral. *Fuzzy Sets and Systems*, 151(2):237–259, 2005.

[GL10] Michel Grabisch and Christophe Labreuche. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286, 2010.

[GLTW08] Judy Goldsmith, Jérôme Lang, Miroslaw Truszczynski, and Nic Wilson. The computational complexity of dominance and consistency in CP-nets. *Journal of Artificial Intelligence Research*, 33:403–432, 2008.

[GMPU06] Aristides Gionis, Heikki Mannila, Kai Puolamäki, and Antti Ukkonen. Algorithms for discovering bucket orders from data. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, KDD 2006*, pages 561–566. ACM, 2006.

[GMS08] Salvatore Greco, Vincent Mousseau, and Roman Słowiński. Ordinal regression revisited: multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191(2):416–436, 2008.

[GP04] Christophe Gonzales and Patrice Perny. GAI networks for utility elicitation. In *Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning, KR2004*, volume 4, pages 224–234, 2004.

[GPD11] Christophe Gonzales, Patrice Perny, and J Ph Dubus. Decision making with multiple objectives using GAI networks. *Artificial Intelligence*, 175(7-8):1153–1179, 2011.

[GPQ08] Christophe Gonzales, Patrice Perny, and Sergio Queiroz. GAI-networks: Optimization, ranking and collective choice in combinatorial domains. *Foundations of Computing and Decision Sciences*, 33(1):3–24, 2008.

[GR08] Michel Grabisch and Eric Raufaste. An empirical study of statistical properties of the Choquet and Sugeno integrals. *IEEE Transactions on Fuzzy Systems*, 16(4):839–850, 2008.

[GRW15] Anne-Marie George, Abdul Razak, and Nic Wilson. The comparison of multi-objective preference inference based on lexicographic and weighted average models. In *Proceedings of the 27th International Conference on Tools with Artificial Intelligence, ICTAI 2015*, pages 88–95. IEEE, 2015.

[HCL+03] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. `https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`, 2003. [Online; accessed 9-May-2008].

[HGO99] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Support vector learning for ordinal regression. In *International Conference on*

Artificial Neural Networks, pages 97–102, 1999.

[HGO00] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[HHR+03] Peter Haddawy, Vu Ha, Angelo Restificar, Benjamin Geisler, and John Miyamoto. Preference elicitation via theory refinement. *Journal of Machine Learning Research*, 4:317–337, December 2003.

[HKLL+06] Mimi Hwang, James Kemp, Eva Lerner-Lam, Nancy Neuerburg, and Paula Okunieff. Advanced public transportation systems: State of the art update 2006. Technical report, The U.S. Federal Transit Administration, 2006.

[HMG07] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: a bayesian skill rating system. In *Advances in Neural Information Processing Systems, , NIPS 2007*, pages 569–576, 2007.

[HSS14] Toby Dylan Hocking, Supaporn Spanurattana, and Masashi Sugiyama. Support vector comparison machines. *arXiv preprint arXiv:1401.8008*, 2014.

[HW12] Wesam Mohamed Herbawi and Michael Weber. A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows. In *Proceedings of The 14th Annual Conference on Genetic and Evolutionary Computation*, pages 385–392. ACM, 2012.

[JD88] Anil K Jain and Richard C Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., 1988.

[JGP+05] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161. ACM, 2005.

[JK02] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[JLS82] Eric Jacquet-Lagrèze and Jean Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the UTA method. *European Journal of Operational Research*, 10(2):151–164, 1982.

[Joa02] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2002*, pages 133–142, New York, NY, USA, 2002. ACM.

[JS09] Tony Jebara and Pannagadatta K Shivaswamy. Relative margin machines. In *Advances in Neural Information Processing Systems, NIPS 2009*, pages 1481–1488, 2009.

[Kau98] Linda Kaufman. Solving the quadratic programming problem arising in support vector classification. *Advances in Kernel Methods-Support Vector Learning*, pages 147–167, 1998.

[KGS12] Miłosz Kadziński, Salvatore Greco, and Roman Słowiński. Selection of a representative value function in robust multiple criteria ranking and choice. *European Journal of Operational Research*, 217(3):541–553, 2012.

[KHM05] Hideto Kazawa, Tsutomu Hirao, and Eisaku Maeda. Order SVM: a kernel method for order learning based on generalized order statistics. *Systems and Computers in Japan*, 36(1):35–43, 2005.

[Kir08] Wilhelm Kirch. Level of measurement. In *Encyclopedia of Public Health*, pages 851–852. Springer Netherlands, Dordrecht, 2008.

[Koj07] Ivan Kojadinovic. A weight-based approach to the measurement of the interaction among criteria in the framework of aggregation by the bipolar Choquet integral. *European Journal of Operational Research*, 179(2):498–517, 2007.

[KP08] Souhila Kaci and Henri Prade. Mastering the processing of preferences by using symbolic priorities in possibilistic logic. In *Proceedings of the 18th European Conference on Artificial Intelligence, ECAI 2008*, pages 376–380, 2008.

[KR93] Ralph L Keeney and Howard Raiffa. *Decisions with multiple objectives: preferences and value trade-offs*. Cambridge University Press, 1993.

[KRR79] Ralph L. Keeney, Howard Raiffa, and David W. Rajala. Decisions with multiple objectives: Preferences and value trade-offs. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(7):403–403, July 1979.

[KT03] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. In *ACM SIGIR Forum*, volume 37, pages 18–28. ACM, 2003.

[LS82] Graham Loomes and Robert Sugden. Regret theory: An alternative theory of rational choice under uncertainty. *The Economic Journal*, 92(368):805–824, 1982.

[LYP10] Yuh-Jye Lee, Yi-Ren Yeh, and Hsing-Kuo Pao. An introduction to support vector machines. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.488.5833&rep=rep1&type=pdf`, 2010. [Online; accessed 9-May-2018].

[Mei03] Marina Meila. Data centering in feature space. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, AISTATS 2003, Key West, Florida, USA, January 3-6, 2003*, 2003.

[MGDJ+12] David G Méndez, Jose L Girela, Joaquin De Juan, M Jose Gomez-Torres, and Magnus Johnsson. Predicting seminal quality with artificial intelligence methods. *Expert Systems with Applications*, 39(16):12564–12573, 2012.

[ML05] Jae H Min and Young-Chan Lee. Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems With Applications*, 28(4):603–614, 2005.

[MRW13] Radu Marinescu, Abdul Razak, and Nic Wilson. Multi-objective constraint optimization with tradeoffs. In *Proceedings of the 19th International Conference on Principles and Practice of Constraint Programming, CP 2013, Uppsala, Sweden, September 16-20, 2013*, pages 497–512. Springer Berlin Heidelberg, 2013.

[NKNW96] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied Linear Statistical Models*, volume 4. Irwin Chicago, 1996.

[NP06] Constantin Niculescu and Lars-Erik Persson. *Convex functions and their applications: a contemporary approach*. Springer Science & Business Media, 2006.

[OFG97] Edgar Osuna, Robert Freund, and Federico Girosi. An improved training algorithm for support vector machines. In *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing*, pages 276–285. IEEE, 1997.

[OK89] Muhittin Oral and Ossama Kettani. Modelling the process of multiattribute choice. *Journal of the Operational Research Society*, 40(3):281–291, 1989.

[O'M13] Conor O'Mahony. *Reasoning with Sorted-Pareto Dominance and Other Qualitative and Partially Ordered Preferences in Soft Constraints*. PhD thesis, University College Cork, 2013.

[OW12] Conor O'Mahony and Nic Wilson. Sorted pareto dominance: An extension to pareto dominance and its application in soft constraints. In *Proceedings of the 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012*, volume 1, pages 798–805. IEEE, 2012.

[OW13] Conor O'Mahony and Nic Wilson. Sorted-pareto dominance and qualitative notions of optimality. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 449–460. Springer, 2013.

[Par71] Vilfredo Pareto. *Manual of Political Economy*. Macmillan, 1971.

[PMP+08] Rajul Parikh, Annie Mathai, Shefali Parikh, G Chandra Sekhar, Ravi Thomas, et al. Understanding and using sensitivity, specificity and predictive values. *Indian Journal of Ophthalmology*, 56(1):45, 2008.

[Raw09] John Rawls. *A Theory of Justice*. Harvard University Press, 2009.

[Raz14] Abdul Razak. *Reasoning with Imprecise Trade-Offs in Decision Making Under Certainty and Uncertainty*. PhD thesis, University College Cork, 2014. `https://cora.ucc.ie/handle/10468/1922`.

[RJ05] Filip Radlinski and Thorsten Joachims. Query chains: learning to rank from implicit feedback. In *Proceedings of the Eleventh ACM*

SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD 2005, pages 239–248. ACM, 2005.

[RVW04] Francesca Rossi, Kristen Brent Venable, and Toby Walsh. mCP nets: representing and reasoning with preferences of multiple agents. In *Proceedings of the 19th National Conference on Artifical Intelligence, AAAI 2004*, volume 4, pages 729–734, 2004.

[RVW11] Francesca Rossi, Kristen Brent Venable, and Toby Walsh. A short introduction to preferences: between artificial intelligence and social choice. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(4):1–102, 2011.

[RW10] S Ravi and S Wilson. Face detection with facial features and gender classification based on support vector machine. *International Journal of Imaging Science and Engineering*, pages 23–28, 2010.

[Saa88] Thomas L Saaty. What is the analytic hierarchy process? In *Mathematical Models for Decision Support*, pages 109–121. Springer, 1988.

[Saa03] Thomas L Saaty. Decision-making with the AHP: Why is the principal eigenvector necessary. *European Journal of Operational Research*, 145(1):85–91, 2003.

[Saa05] Thomas L Saaty. The analytic hierarchy and analytic network processes for the measurement of intangible criteria and for decision-making. *Multiple Criteria Decision Analysis: State of The Art Surveys*, pages 345–405, 2005.

[Sar06] Jennifer Saranow. Carpooling for grown-ups—high gas prices, new services give ride-sharing a boost; rating your fellow rider. *Wall Street Journal*, Feb 2006.

[Sch86] David Schmeidler. Integral representation without additivity. *Proceedings of The American Mathematical Society*, 97(2):255–261, 1986.

[Sch03] Robert E Schapire. The boosting approach to machine learning: An overview. In *Nonlinear Estimation and Classification*, pages 149–171. Springer, 2003.

[Sen70] Amartya Sen. *Collective Choice and Social Welfare*. Mathematical Economics Texts Series. Holden-Day, 1970.

[SGM05] Yannis Siskos, Evangelos Grigoroudis, and Nikolaos F Matsatsinis. UTA methods. In *Multiple Criteria Decision Analysis: State of The Art Surveys*, pages 297–334. Springer, 2005.

[SGMP18] Olivier Sobrie, Nicolas Gillis, Vincent Mousseau, and Marc Pirlot. UTA-poly and UTA-splines: additive value functions with polynomial marginals. *European Journal of Operational Research*, 264(2):405–418, 2018.

[Sho83] Anthony F Shorrocks. Ranking income distributions. *Economica*, 50(197):3–17, 1983.

[SHS01] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

[SKF08] Andreas Stolcke, Sachin Kajarekar, and Luciana Ferrer. Nonparametric feature normalization for SVM-based speaker verification. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2008, March 30 - April 4, 2008, Caesars Palace, Las Vegas, Nevada, USA*, pages 1577–1580. IEEE, 2008.

[Smi15] Aaron Smith. US smartphone use in 2015. Pew Research Center, http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/, 2015.

[SS73] Venkataraman Srinivasan and Allan D Shocker. Linear programming techniques for multidimensional analysis of preferences. *Psychometrika*, 38(3):337–369, 1973.

[SSY99] Yannis Siskos, Athanasios Spyridakos, and Denis Yannacopoulos. Using artificial intelligence and visual techniques into preference disaggregation analysis: The MUDAS system. *European Journal of Operational Research*, 113(2):281–299, 1999.

[SWM93] W Nick Street, William H Wolberg, and Olvi L Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *IS&T/SPIE's Symposium on Electronic Imaging: Science and Technology*, pages 861–870. International Society for Optics and Photonics, 1993.

[SY85] Yannis Siskos and D Yannacopoulos. UTASTAR: An ordinal regression method for building additive value functions. *Investigaçao Operacional*, 5(1):39–53, 1985.

[TD00] DM Tax and Robert PW Duin. Feature scaling in support vector data descriptions. *Learning from Imbalanced Datasets*, pages 25–30, 2000.

[Tes89] Gerald Tesauro. Connectionist learning of expert preferences by comparison training. In *Advances in Neural Information Processing Systems, NIPS 1989*, pages 99–106, 1989.

[TK01] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2(Nov):45–66, 2001.

[UPGM09] Antti Ukkonen, Kai Puolamäki, Aristides Gionis, and Heikki Mannila. A randomized approximation algorithm for computing bucket orders. *Information Processing Letters*, 109(7):356–359, 2009.

[Van99] Robert J Vanderbei. LOQO: An interior point code for quadratic programming. *Optimization Methods and Software*, 11(1-4):451–484, 1999.

[Vap13] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer Science & Business Media, 2013.

[VN59] John Von Neumann. On the theory of games of strategy. *Contributions to the Theory of Games*, 4:13–42, 1959.

[VNM07] John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 2007.

[Wal50] Abraham Wald. *Statistical Decision Functions*. Wiley, 1950.

[Wal07] Toby Walsh. Representing and reasoning with preferences. *AI Magazine*, 28(4):59, 2007.

[Wil04] Nic Wilson. Extending CP-nets with stronger conditional preference statements. In *Proceedings of the Nineteenth National Conference on Artificial Intelegence, AAAI 2004*, volume 4, pages 735–741, 2004.

[WK94] Mirjam RM Westenberg and Pieter Koele. Multi-attribute evaluation processes: Methodological and conceptual issues. *Acta Psychologica*, 87(2):65–84, 1994.

[WM16a] Nic Wilson and Mojtaba Montazery. Preference inference through rescaling preference learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 2203–2209. AAAI Press, 2016.

[WM16b] Nic Wilson and Mojtaba Montazery. *Preference Inference Through Rescaling Preference Learning (extended version of IJCAI 2016 paper including proofs)*. Available at http://ucc.insight-centre.org/nwilson/RescalingProofs.pdf, 2016.

[WN06] Stephan Winter and Silvia Nittel. Ad hoc shared-ride trip planning by mobile geosensor networks. *International Journal of Geographical Information Science*, 20(8):899–916, 2006.

[WRM15] Nic Wilson, Abdul Razak, and Radu Marinescu. Computing possibly optimal solutions for multi-objective constraint optimisation with tradeoffs. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pages 815–821. AAAI Press, 2015.

[XC08] Lirong Xia and Vincent Conitzer. Determining possible and necessary winners under common voting rules given partial orders. In *Proceedings of the 23rd National Conference on Artificial Intelligence, AAAI 2008*, pages 196–201. AAAI Press, 2008.

[XCS06] Linli Xu, Koby Crammer, and Dale Schuurmans. Robust support vector machine training via convex outlier ablation. In *Proceedings of The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, volume 6, pages 536–542, 2006.

[YMH09] Georgios N Yannakakis, Manolis Maragoudakis, and John Hallam. Preference learning for cognitive modeling: a case study on entertainment preferences. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39(6):1165–1175, 2009.

[YSW07] Xulei Yang, Qing Song, and Yue Wang. A weighted support vector machine for data classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(05):961–976, 2007.

[YV13] N Yuvaraj and P Vivekanandan. An efficient SVM based tumor classification with symmetry non-negative matrix factorization using gene expression data. In *Proceedings of the 2013 International Conference on Information Communication and Embedded Systems, ICICES 2013*, pages 761–768. IEEE, 2013.

[YWL+11] Fusun Yaman, Thomas J Walsh, Michael L Littman, et al. Democratic approximation of lexicographic preference models. *Artificial Intelligence*, 175(7-8):1290–1307, 2011.

[YYT09] I-Cheng Yeh, King-Jang Yang, and Tao-Ming Ting. Knowledge discovery on RFM model using Bernoulli sequence. *Expert Systems with Applications*, 36(3):5866–5871, 2009.

[ZCSZ07] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 287–294. ACM, 2007.

[Zic12] Kathryn Zickuhr. Three-quarters of smartphone owners use location-based services. Pew Research Center, http://www.pewinternet.org/2012/05/11/three-quarters-of-smartphone-owners-use-location-based-services/, 2012.