Noname manuscript No.
(will be inserted by the editor)

Using Negotiable Features for Prescription Problems

Antonio Bella · Cèsar Ferri · José Hernández-Orallo · María José Ramírez-Quintana

Received: date / Accepted: date

Abstract Data mining is usually concerned on the construction of accurate models from data, which are usually applied to well-defined problems that can be clearly isolated and formulated independently from other problems. Although much computational effort is devoted for their training and statistical evaluation, model deployment can also represent a scientific problem, when several data mining models have to be used together, constraints appear on their application, or they have to be included in decision processes based on different rules, equations and constraints. In this paper we address the problem of combining several data mining models for objects and individuals in a common scenario, where not only we can affect decisions as the result of a change in one or more data mining models, but we have to solve several optimisation problems, such as choosing one or more inputs to get the best overall result, or readjusting probabilities after a failure. We illustrate the point in the area of Customer Relationship Management (CRM), where we deal with the general problem of prescription between products and customers. We introduce the concept of negotiable feature, which leads to an extended taxonomy of CRM problems of greater complexity, since each new negotiable feature implies a new degree of freedom. In this context, we introduce several new problems and techniques, such as data mining model inversion (by ranging on the inputs or by changing classification problems into regression problems by function inversion), expected profit estimation and curves, global optimisation through a Montecarlo method, and several negotiation strategies in order to solve this maximisation problem.

This work has been partially supported by the EU (FEDER) and the Spanish MEC/MICINN, under grant TIN 2007-68093-C02, the Spanish project "Agreement Technologies" (Consolider Ingenio CSD2007-00022) and the GVA project PROMETEO/2008/051.

DSIC-ELP, Universidad Politécnica de Valencia, Camí de Vera s/n, 46022 Valencia (Spain)
E-mail: {abella,cferri,jorallo,mramirez}@dsic.upv.es

Keywords: data mining, profit maximisation, function inversion problem, global optimisation, negotiation, CRM, ranking, probability estimation, negotiable features, Montecarlo method.

1 Introduction

Complex decisions are characterised by the search of a trade-off among a set of constraints, models, rules and equations, where several computational techniques (linear programming, simulation, numerical computation, operational research, etc.) can be used to solve the optimisation problem. More and more frequently, models are not derived by experts (from the business or scientific domain involved) but automatically inferred by data mining techniques. In this context, many optimisation techniques are no longer valid, since models are usually expressed in a non-mathematical way (e.g. decision tree) or even as a black-box (e.g. neural networks). Consequently, many techniques are no longer valid because the mathematical properties (continuity, monotonicity, ...) of the functions which describe the data mining models are unknown.

As a result of all this, the combination of data mining and (global) optimisation has recently received an increasing attention [6][22][20]. Much of this work originates from real application problems which appear in the area of Customer Relationship Management (CRM) [4][3]. CRM is an application field where econometrics and mainstream data mining can merge, along with techniques from simulation, operational research, artificial intelligence and numerical computation.

Decisions in the context of prescription problems deal about distinguishing or ranking the products to be offered to each customer (or, symmetrically, selecting the customers to whom we should make an offer), establishing the moment or sequence of the offers, and determining the price, warranty, financing or other associated features of products and customers. The classical application of data mining for prescription problems has usually considered a rather monolitic and static view of the process, where we have one or more products to be offered to a pool of customers, and we need to determine a sequence of offers (product, customer) to maximise profit. These and related problems (e.g. cross-selling or up-selling) have been addressed with techniques known as "mailing/selling campaign design" [3] or from the more general view of recommender systems [1], which are typically based on data mining models which perform good rankings and/or good probability estimations.

However, in more realistic and interactive scenarios, we need to consider that a better customisation has to be performed. It is not only the choice of products or customers which is possible, but several features of the product (or the deal) can be tuned or adapted to get more earnings. We call these features 'negotiable features', and are common in everyday applications: price, delivery time, payment method, bonuses, warranties, etc. This suggests a negotiation process over these features that we want to be automated and optimal on the side of the seller.

The consequence of this more general and interactive view is that the already complex prescription problem becomes much more difficult since we have much more variables (degrees of freedom), which have to be taken into account in the optimisation problem. And the key issue is that the relation between all these variables (price, delivery time, payment method, bonuses, warranties, etc.) and the dependent variable (e.g. buying or not) is the data mining model, which can be incarnated in the form of decision trees, neural networks, support vector machines, linear, non-linear or logistic regression, nearest neighbours, etc. Of course we could devise a specific method to handle this when the model is, e.g., a logistic regressor, but, in general, we would like to be able to use the whole range of techniques which are available in the data mining suite at hand. That means that we have to treat the models as black boxes, not assuming any particular property. In this way, the strategies we present here are general and do not need to be adapted to each type of learning algorithm.

The notion of negotiable feature in data mining models as it is presented here is new in the literature. The term appears scarcely in the literature but with other meanings [33][8]. In these works, the focus has been set on agent negotiation strategies using a traditional data mining presentation, where the specific relation between the feature and the output is not obtained. We propose to consider negotiable features as input variables for a "general function inversion problem", i.e., given a function, calculate the input combinations which produce a given output. For instance, if we have a model which gives us the probability of buying a product by a customer, we would like to use the model to obtain the possible pairs (or ranges of price and delivery time) such that the result is a probability of 0.75 of the customer buying the product.

As a consequence of being able to adjust and customise some features of the offer, it is expected that many different offers for the same product and customer could be made, so, naturally, a negotiation process appears, where seller and buyer can make bids and counteroffers. The rejection of an offer entails a recomputation of the probabilities, something which must be done on the data mining model.

The previous new family of open problems, which are common in CRM, are also usual in other areas, where prescription has to be made. For instance, medical prescription is another area where drugs and treatments have to be customised. In this case, the "class" is not whether a patient will or not buy a drug, but whether the treatment will work or not. The agents of negotiation here are not the doctor and the patient, but the doctor and the illness. Data mining models are about treatment effectivenes, and the profit is changed into a more complex evaluation of treatment costs, counterindications and recovery periods. Similarly, other areas such as education (teacher-student), social networks (recommendations), human resources (hiring), etc., can be understood and solved under the paradigm we present in this work.

The major contributions of this work are then:

- A taxonomy of prescription problems embracing the classical static problems without negotiable features and the new interactive problems which include four new open problems depending on the number of kinds of items and customers.
- The notion of negotiable feature, its formal definition and properties, which generalises prescription problems as negotiation problems.
- The study of the function inversion problem for data mining models which, in some cases, can turn a classification problem into a regression problem (or viceversa).
- The notion of profit curves derived from probabilistic models, their normalisation after a bid and the use of envelope curves to compare and combine several curves.
- Several new strategies for negotiation using data mining models, some of them shown to be better than the baseline methods, especially the one based on a global view of the negotiation.

A real scenario is used in the paper. In this scenario, a real estate agent has a pool of housings to sell and a pool of customers. We have performed a thorough series of experiments which show that the way in which the model is deployed and applied is crucial to get good results. In fact, only with the set of techniques developed in this paper, the results using data mining models surpass the results of models based on an increment over the average (baseline methods).

The paper is organised as follows. In Section 2, we quickly survey the previous work on prescription problems and highlight a new family of prescription problems which have not been addressed to date, those for which negotiable features appear. The classical prescription problems and the new ones conform a new taxonomy which is presented in this section. Section 3 introduces the notion of negotiable feature in data mining models, its formalisation and properties. In Section 4, we analyse the use of these properties to enhance data mining models and we study several options for the inversion problem that takes place when we want to determine the value for a negotiable feature given a probability or a threshold for the output. Section 5 introduces three new different negotiation strategies (maximum expected profit, best local expected profit and maximum global optimisation) and the new scenarios derived from the taxonomy. In Section 6, we analyse our methods with real data on our example domain (real estate agent's) and compare the total profit for each method and negotiation strategy on several of the new scenarios introduced in the taxonomy. In Section 7, we analyse the connections between our work and some previous or related approaches. Section 8 closes the paper with some concluding remarks and the future work.

2 A Taxonomy of Prescription Problems

Prescription problems are a very common type of predictive problems where an item is prescribed to an individual. The item can be a commercial product,

Table 1 Different prescription problems that consider the number of different kinds of products to sell, whether the net price for the product is fixed or negotiable, and the number of customers.

Case	Kinds of products	Features	Number of customers	Approach
1	1	fixed	1	Trivial
2	1	fixed	C	Customer ranking [3]
3	N	fixed	1	Product ranking [3]
4	N	fixed	C	Joint Cut-off [2]
5	1	negotiable	1	-
6	1	negotiable	C	-
7	N	negotiable	1	-
8	N	negotiable	C	-

a drug, a webpage, blog or reading reference, an action, etc., depending on the application. The individual can be a user, a customer, a patient, a student, a robot, etc. Data mining models are used to model the behaviour of each individual in order to assess the appropriateness of each item for each user. Typically, prescription problems are classified according to the number of kinds of items and individuals, since the problem complexity depends on the number of combinations. However, the prescription scenario using data mining models is static, in the sense that once the prescription is made, the item is ruled out and any subsequent prescription is made on other items (this does not mean, of course, that the interaction of previous transactions is not used for subsequent prescriptions or recommendations). In this work we do consider the case that the same item (or the very prescription) can be adapted to the individual. In order to do that, the items must present some adjustable features that we call "negotiable features" (a more precise definition will be given in the following section). If these features exist it is not only the item and the individual that we have to match but also the best values for the negotiable features.

In this section we devise a taxonomy of prescription problems depending on several variables involved in the process. This will help to recognise the previous work in this area and the open problems we aim to solve. Since most of the related work comes from the area of Customer Relationship Management (CRM), from now on, instead of using the general terms 'item' and 'individual', we will use the more specific terms 'product' and 'customer', which are typical in CRM. As usual, we consider the number of customers (C) and the different kinds of products (N). But we also study the presence or absence of negotiation in the transaction. As we have stated in the introduction, if at least one feature is negotiable, then we can introduce some kind of negotiation into the process; however, if all the features are non-negotiable (fixed), then we are dealing with a traditional prescription problem. In all these problems, there is an optimality criterion (or utility function) which shapes the goal. In CRM, the goal is usually the net profit (although other goals such as customer loyalty are possible). In general, other possible criteria might depend on other resources (time, people involved, security, etc.).

Table 1 shows eight different prescription problems that are defined by considering the number of products and customers involved as well as the fixed or negotiable nature of the features of each product. The last column

shows several approaches that have already been proposed in the literature for solving some of these problems. The rows with a "-" in this column indicate cases that are (first) addressed in this paper. We discuss each of them in more detail below.

2.1 Case with one kind of product, fixed features, and one customer

Case 1 in Table 1 is trivial. In this scenario, the seller offers the product to the customer with fixed conditions/features and the customer may or not buy the product. The seller cannot do anything more because s/he does not have more products to sell. S/he cannot negotiate the price, warranty, delivery time, etc., of the product with the customer, and s/he does not have any more customers for the product.

2.2 Case with one kind of product, fixed features, and C customers

Case 2 in Table 1 is the typical case of a mailing campaign design. The objective is to obtain a customer ranking to determine the set of customers to whom the mailing campaign should be directed in order to obtain the maximum profit. Data mining can help in this situation by learning a probabilistic classification model[1] from previous customer data that includes information about similar products that have been sold to them. This model will obtain the buying probability for each customer. Sorting them by decreasing buying probability, the most desirable customers will be at the top of the ranking. Using a simple formula for marketing costs, we can establish a threshold/cut-off in this ranking. The customers above the threshold will be offered the product. This is usually plotted using the so-called lift charts (see e.g. [3]).

2.3 Case with N kind of products, fixed features, and one customer

Case 3 in Table 1 is symmetric to case 2. Instead of N customers and one product, in this case, there are N different products and only one customer. Hence, the objective is to obtain a product ranking for the customer. Similarly, data-mining can help to learn a probabilistic estimation model from previous product data that have been sold to similar customers. This model will predict the buying probability for each product, so by putting them in order of decreasing buying probability, the most desirable products for the customer will be at the top of the ranking. This case overlaps to a great extent with the typical applications of recommender systems [1], so many techniques can be applied here, although predictive models which show good probability estimation and ranking (typically measured with the AUC, MSE or the logloss, see e.g. [11]) are custom here.

[1] A probabilistic classification model is a model which outputs the probability for the class, e.g. [10].

2.4 Case with N kinds of products, fixed features, and C customers

Case 4 in Table 1 is studied in [2]. This situation is more complex than the cases 2 and 3, since there is a data-mining model for each product. In other words, there are N customer rankings (one for each product) and the objective is to obtain the set of pairs customer-product that gives the maximum overall profit. Note that, normally, the best local cut-off of each model (the set of customers that gives the maximum profit for one product) does not give the best global result. Moreover, several constraints are frequently required in real applications (limited stock of products, the customers may be restricted to only buy one product). Two different methods are proposed in [2] to obtain the global cut-off: one is based on merging the prospective customer lists and using the local cut-offs, and the other is based on simulation. The study in [2] shows that using simulation to set model cut-off obtains better results than classical analytical methods.

2.5 Cases with negotiable features

In this paper, we deal with cases 5, 6, 7 and 8, which, to our knowledge, have not been addressed in the literature. These cases are similar to cases 1, 2, 3 and 4 but, in these cases, at least one feature is negotiable. This represents a complete new scenario that introduces more degrees of freedom in the search space for an optimal solution. Additionally, it usually entails a negotiation process, which usually means an episode of offers and counter-offers from the negotiation parts that makes the whole process more flexible, and logically, more difficult to analyse. Before presenting our approaches to these scenarios, next we formalise the idea of having features that can be modified for a prescription, known as "negotiable features", which make a negotiation process possible.

3 Negotiable Features

As we have already mentioned in previous sections, there are many data mining problems in which one or more input features can be modified at the time the model is applied, turning the problem into some kind of negotiation process. In this section, we formally define the concept of *negotiable feature*, and we discuss which properties are derived from their special nature and how these features can be used.

3.1 Negotiable Feature Definition

Consider a supervised problem, where $f : X_1 \times X_2 \times ... \times X_m \rightarrow Y$ denotes the target (real) function; X_i, $i \in 1..m$ denote input feature (or attribute) domains, and Y denotes the output attribute domain (or class). Values for

input and output attributes will be denoted by lowercase letters. Hence, labelled instances are then tuples of the form $\langle x_1, x_2, ..., x_m, y\rangle$ where $x_i \in X_i$ and $y \in Y$.

We assume that there is a *(non-)strict total order* for the output, i.e., there is a relation $\succ$ such that for every two different possible values $y_a, y_b \in Y$, we have that either $y_a \succ y_b$ or $y_b \succ y_a$. If the order is non-strict[2], we denote it as $\succeq$. This order usually represents some kind of benefit, utility or cost. For numerical outputs, $\succ$ is usually the order relation between real numbers (either $<$ or $>$, depending on whether it is a cost or benefit). For nominal outputs, $\succ$ usually sets an order between the classes. For binary problems, where POS and NEG represent the positive and negative class respectively, we can just set that $POS \succ NEG$. For more than two classes, the order relation can be derived from the cost of each class. For instance, if we have three buying results (buy, does not buy, chums), we can order them by their costs. Note that this does not mean that classes are necessarily ordinal (such as e.g. $\{low, medium, high\}$). Analogously, we also assume there is a *non-strict total order relation* for each input attribute X_i denoted as $\succeq_i$. For readability, in what follows we will omit the subscript when it is clear from the context.

With these definitions, we can now formalise the idea of negotiable feature.

Definition 1 (Negotiable Feature for an instance)
An attribute X_i is said to be a negotiable feature (or attribute) for an instance $\langle x_1, x_2, ..., x_m, y\rangle$ iff

1. (adjustability) The values for X_i can be varied at model application time.
2. (sensitivity) Fixing the values of all the other input attributes $X_j \neq X_i$ of the instance, there are two different values for X_i producing different output values.
3. (monotonicity) The relation between the input feature X_i and the output Y is either
   - monotonically increasing: for any two values $a, b \in X_i$, if $a \succeq b$ then $f(x_1, x_2, ..., a, ..., x_m) \succeq f(x_1, x_2, ..., b, ..., x_m)$
     or
   - monotonically decreasing: for any two values $a, b \in X_i$, if $a \succeq b$ then $f(x_1, x_2, ..., a, ..., x_m) \preceq f(x_1, x_2, ..., b, ..., x_m)$.

Both conditions 2 and 3 define a *monotonic dependency* between the negotiable attribute and the output.

Based on the previous definition, we introduce the concept of negotiable feature for a problem.

Definition 2 (Negotiable Feature in a problem)
Let D be a set of instances defining a supervised problem f. We will say that a feature is a negotiable feature in f, if conditions 1 and 3 hold for all the instances in D, and 2 holds for a significant/relevant proportion $0 < \tau \leq 1$ of instances in D.

[2] As usual, a strict total order relation can be defined from a non-strict total order.

This value τ is determined by the user depending on the problem, the presence of noise, etc. Note that the relaxation of condition 2 (sensitivity) is due to the fact that, especially in classification problems, for some instances, the output cannot be changed by only changing the value of the negotiable attribute, i.e., the attribute cannot be negotiable for some instances (but it is for the remaining). For example, if someone cannot pay more than 500 euros per year for a life insurance, he will not sign a 1000-euro life insurance contract even if he is offered an extra accident coverage. In some case, we can have dynamic negotiable features, i.e., features that were not negotiable initially, but a special condition or charge in other attributes can make these features negotiable. For example, in the airline and railway industry, a drop in the price might make the warranty-feature negotiable.

Next, we give an example of negotiable and non-negotiable features.

Example 1 In a loan granting model, where loans are granted or not according to a model which has been learnt from previous customer behaviours, the age of the customer is not a negotiable feature, since we cannot modify it (condition 1 is violated). The bank branch office where the contract can take place is also an input, which we can modify, but it is not a negotiable feature either since it rarely affects the output (property 2 is violated). The number of meetings is also modifiable and it frequently affects the result, but it is usually non-monotonic, so it is not a negotiable feature either (property 3 is violated). In contrast, the loan amount, the interest rate or the loan period are negotiable features since very large loan amounts, very low interest rates and very long loan periods make loans unfeasible for the bank.

The definition of an order relation for inputs and outputs is usually straightforward and it allows different tasks to be seen in a uniform way. For instance, we can have the following possibilities depending on the nominal or numerical character of the input and output.

- Nominal negotiable feature and nominal output. In this classification task, there is a relation between some particular values of the negotiable attribute and some values of the class. For example, a travel agency survey will be negative whenever the traveller finds overbooking at the airport. Negotiation can take place if we ensure that no overbooking will take place.
- Numerical negotiable feature and nominal output. In this classification task, there is a relation between the range of values of the negotiable attribute and some values of the class. The loan amount, the interest rate and the loan period are examples of negotiable features for the loan problem.
- Nominal negotiable feature and numerical output. In this regression task, there is a relation between some particular values of the negotiable attribute and the range of the output. For instance, the time for a delivery monotonically depends on whether we use regular mail or fast courier. This feature can be a negotiable feature.
- Numerical negotiable feature and numerical output. In this regression task, there is a relation between the range of values of the negotiable attribute

and the range of the output. For example, the time for a cargo train trip monotonically depends on the number of station stops or the cargo load.

Although we only show the four possible cases for supervised problems, the first two cases are extensible to clustering, since if we have a model, we can understand groups as classes, and use it as a classification model.

3.2 Negotiable Feature Properties

Given the previous definitions, we can analyse what we can do with negotiable features and how we can exploit their special character.

The first interesting property is that, since they are monotonic and sensitive (in general), i.e., they are monotonically dependent, the inversion problem is possible. This means that questions such as: "tell me the maximum amount for the loan" or "tell me the maximum number of items such that the order can be delivered in one week" or "tell me the price such that there is a 0.75 probability of selling" or "tell me the dose such that the blood pressure goes down to 12" are not only sensible and useful, but solvable too. In the next section we will explore some possible solutions.

The second interesting property is also directly derived from monotonicity, and is related to the idea that negotiable features have a negotiation direction, such that once reached a minimum or maximum output, it is useless to go on negotiating on that feature. For instance, if we can grant a loan for 200,000 euros we can also grant a loan for 150,000 euros. It is especially advantageous for classification problems, but it also happens for many regression problems.

In order to clarify this implication, consider that we have a minimum or maximum (or both) for the output feature, using the derived order. The maximum is just defined as $max(Y) = y_i$ such that for all j we have that $y_i \succeq y_j$. Similarly we can define the minimum. For numerical outputs (i.e., regression), we usually have one of them, e.g. minimum price, minimum delivery time, etc., but we can also have both (e.g. minimum and maximum load). For nominal outputs, since the number of classes is finite, we always have a minimum and a maximum. In binary classification problems, the maximum is class POS and the minimum is class NEG. With this definition, we can show the following results:

Proposition 1 *Consider that there exists a maximum value for the output feature Y, denoted by y_{max}, and there is a negotiable feature X_i which is monotonically increasing (respectively monotonically decreasing) wrt. Y.*

If $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) = y_{max}$ then for every b, such that $b \succeq a$ (respectively $a \succeq b$) we also have that $f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) = y_{max}$.

Proof If the relation is monotonically increasing, we have that for any two values a, b for X_i, if $b \succeq a$ then

$$f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) \succeq f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m)$$

Since $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) = y_{max}$ we have that

$$f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) \succeq y_{max}$$

but since y_{max} is the maximum, then $f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) = y_{max}$. If the relation is monotonically decreasing the proof is similar.

Proposition 2 *Consider that there exists a minimum value for the output feature Y, denoted by y_{min}, and there is a negotiable feature X_i which is monotonically increasing (respectively monotonically decreasing) wrt. Y.*

If $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) = y_{min}$ then for every b, such that $a \succeq b$ (respectively $b \succeq a$) we also have that $f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) = y_{min}$.

Proof The proof is similar to the previous proposition.

Let us see an example.

Example 2 Following with the loan problem, the loan amount (denoted by δ) is a negotiable attribute which is monotonically decreasing wrt. the class (POS means granting the loan and NEG means not granting it). In this case, $\succeq$ for the negotiable attribute is the standard order relation for real numbers $\geq$. For the class we have that $POS \succ NEG$. Consequently, from the previous propositions, we can derive the following rules for this case:

- If $f(x_1, \ldots, x_{i-1}, \delta_a, x_{i+1}, \ldots, x_m) = POS$ then for every δ_b, such that $\delta_a \geq \delta_b$ we also have that $f(x_1, \ldots, x_{i-1}, \delta_b, x_{i+1}, \ldots, x_m) = POS$, which means that, for a given customer, if we can grant him a loan for a quantity δ_a we can also grant him a loan for a lower quantity.
- If $f(x_1, \ldots, x_{i-1}, \delta_a, x_{i+1}, \ldots, x_m) = NEG$ then for every δ_b, such that $\delta_b \geq \delta_a$ we also have that $f(x_1, \ldots, x_{i-1}, \delta_b, x_{i+1}, \ldots, x_m) = NEG$, which means that, for a given customer, if we cannot grant him a loan for a quantity δ_a we cannot grant him a loan for a higher quantity either.

These rules derived from the previous two propositions will be important in the following approaches to address negotiable feature models and also to derive negotiation strategies, as we will see below in Section 4.

4 Approaches to Negotiable Feature Models

In this section we will explore three possible data mining approaches to obtain and use models in presence of negotiable features. We will focus on classification models where the negotiable feature is numerical and continuous, since this case is very frequent and its inversion problem is the most difficult one (we must go from a small set of output categories to an infinite set of input values). In the case of discrete numerical values (e.g. a discrete number of items) the model prediction (or, more precisely, the chosen value by the negotiation strategy) must be rounded up or down to the closest integer value.

Additionally, we will consider the classification model as a probabilistic model, i.e., a model which outputs probability estimations for the classes.

We will present three general approaches that use negotiable features. At first, traditional approach will be to treat the problem as a classical classification problem. A second approach is similar but using the rules derived from the negotiable feature properties in order to generate more examples and make learning easier. Finally, a third approach transforms the classification problem into a regression one, inverting the original problem.

4.1 Traditional Approach by Using the Monotonicity Property

The first approach just learns a model as usual, by using the available labelled examples to train it, and then uses this model for predicting the output value of future cases.

However, this approach has two drawbacks: (1) the inverse problem (that is, to obtain the value of the negotiable attribute for which a certain output value is obtained) must be solved by iteratively checking different values for the negotiable feature and just selecting the one which produces the desired output, and (2) in many situations we only have examples of one class available.

The first problem can be minimised since we know that the relation between the negotiable feature and the output is monotonic. This means that instead of making an exhaustive search, we only need a binary or gradient search.

The second problem, however, is more dangerous, since in many cases it precludes from learning a useful model. Consider again the loan granting problem presented in Examples 1 and 2. It is usually the case that the bank only records past granted loans. If we use this information as it is, this means that we only have examples from the positive class, so learning is impossible (at least as a supervised problem). Usually, this is minimised by slightly changing the meaning of the class. Instead of using whether a loan has been granted or not, we change it into whether a past granted loan has been or not benefitial for the bank. But even with this typical transformation, we have a quite biased apriori distribution (a type of selection bias known as sample bias [15]), since many possible good and bad customers who did not get the loan have been ruled out from the dataset.

So, in this case, asking questions such as "what is the maximum loan amount we can grant to this customer?" or "which loan amount places this operation at a probability of 0.95 of being a profitable customer?" are more difficult to answer and more dangerous. Precisely, these questions fly around the limit between the information we have in the records and the information we do not have. Typically, in order to answer the previous questions we can draw a curve with the probability of being a profitable customer against the possible values of the negotiable feature. But part of the curve has no supporting evidence. Consequently, models constructed in this way are expected to behave bad for these questions.

4.2 Traditional Approach by Deriving More Examples

According to the second problem formulated in the previous section, and the rules that we can obtain by applying Propositions 1 and 2, we propose a new way of generating more examples which can be used to learn the classification model in a traditional approach. If we recall Proposition 1 we have that: If $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) = y_{max}$ then for every b, such that $b \succeq a$ (respectively $a \succeq b$) we also have that $f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m) = y_{max}$. That means that if we have an example in the training set as: $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) = y_{max}$ we can generate as many *new examples* just changing a for as many b as we like, just that $b \succeq a$ (respectively $a \succeq b$). And the same for the other property.

This means that for binary problems, we can always use one of the two properties and convert one example into hundreds or thousands of examples. In this way, we introduce some knowledge about the relation between the negotiable feature and the output in the training dataset and, as a consequence, into the learnt model. Additionally, this generation can be done in such a way that we can compensate the apriori class distribution bias. Consequently, probability estimation can now take advantage of much more examples, much more information and, hopefully, a less biased dataset.

4.3 Inverting Problem Presentation

A quite different approach is to think about the problem as an inversion problem from scratch. Imagine a model which estimates the delivery time for an order depending on the kind of product and the units which are ordered. One possible (traditional) use of this model is to predict the delivery time given a new order. However, another use of this model is to determine the number of units (provided it is possible to play with this value) that can be delivered in a fixed period of time, e.g. one week. This is an example of an "inverse use" of a data mining model, where all inputs except one and the output are fixed, and the objective is to determine the remaining input value.

Definition 3 (**Inversion problem**) Given a supervised problem $f : X_1 \times X_2 \times ... \times X_i \times ... \times X_m \rightarrow Y$, where X_i is the negotiable feature, the inversion problem consists in defining the function $f^I : X_1 \times \ldots \times X_{i-1} \times Y \times X_{i+1} \times \ldots \times X_m \rightarrow X_i$.

In the above example, f is the function that calculates the delivery time of an order, the negotiable feature X_i is the number of delivered units and f^I calculates this number by considering the delivery time fixed. As mentioned in Section 3.2, the conditions of monotonicity and sensitivity that are satisfied by the negotiable attributes, will enable us to solve this problem, as we explain below.

The inversion problem is well-known [9] and seems simple at first sight, but many questions arise. First, is f^I also a function? In other words, for

two different values for X_i we may have the same value for Y which will ultimately translate into two inconsistent examples for f^I (two equal inputs giving different outputs). Second, the fact that we have an example saying that a given loan amount was granted to a customer does not mean that this is the maximum amount that could be granted to the customer. Third, deriving probabilities to answer questions such as "which loan amount places this operation at a probability of 0.95 of being a profitable customer?" seem to be unsolvable with this new presentation.

Although, it may seem hard to overcome these problems, looking at these issues more carefully, we see that there is still a possible solution which is to consider the inverting problem as a regression one. This is so because, first, many regression techniques work well for inconsistent examples, so this question is not actually a big practical problem. Second, it is true that cases do not represent the maximum amount, but in many cases the examples represent deals and they are frequently not very far away from the maximum. Or, in the worst case, we can understand the new task as "inferring" the typical value for X_i such that the loan is granted to the customer. And third, we can also obtain probabilities in a regression problem.

Then, if we invert the problem, how can we address the original problem again? With the original model and for only two classes, it can be done by calculating $p(POS|\langle x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m\rangle)$, for any possible value $a \in X_i$. From the inverted (regression) problem, we get a prediction:

$$\hat{a} = f^I(x_1, \ldots, x_{i-1}, POS, x_{i+1}, \ldots, x_m)$$

If we think of $\hat{a}$ as the predicted maximum or minimum for a which makes a change on the class, a reasonable assumption is to give 0.5 probability for this point, namely $p(POS|\langle x_1, \ldots, x_{i-1}, \hat{a}, x_{i+1}, \ldots, x_m\rangle) = 0.5$.

The next step is to assume that the output for f^I follows a distribution with centre at $\hat{a}$. For instance, we can assume a normal distribution with mean at $\hat{a}$ and use the relative error (ρ) (on the training set) multiplied by $\hat{a}$, as standard deviation σ. In other words, we use $N(\hat{a}, \rho * \hat{a})$. Our assumption that the values of the negotiable attribute can be modelled by a normal distribution is a working hypothesis which allows us to derive the probabilities in an almost direct way. There are, of course, other alternative ways of estimating the distribution parameters by using a relative squared error as variance or we could use techniques such as bootstrapping. Note that we estimate a different standard deviation for each example (since this is relative to the predicted value $\hat{a}$). It is difficult to get a reliable and specific estimation for each example in that, assuming the use of any particular kind of data mining techniques, since there are many methods which do not output a reliability or expected error for each instance.

From here, we can derive the probability for any possible value a as the cumulative distribution function derived from the above normal, i.e., $\Phi_{\hat{a},\rho*\hat{a}}$.

Figure 1 shows an example of a normal distribution with centre at $\hat{a} = 305,677.9$ and standard deviation $\sigma = 59,209.06$ and its associated cumulative distribution function.

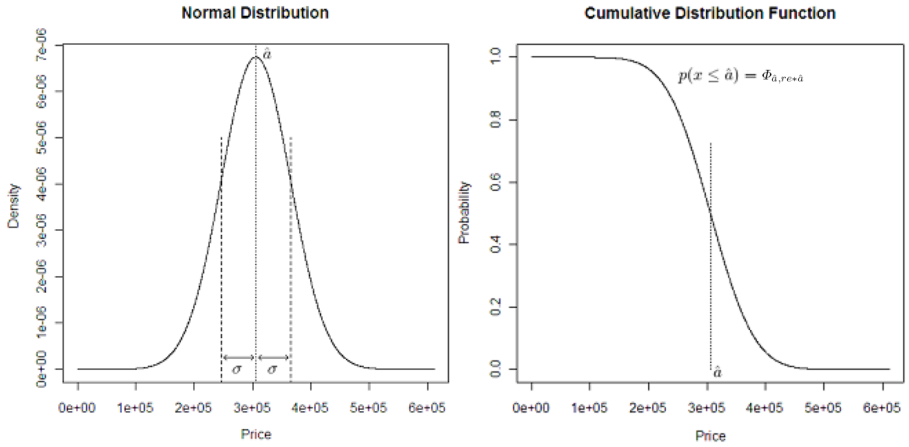

Fig. 1 Left: Example of a normal distribution $\hat{a} = 305,677.9$ and $\sigma = 59,209.06$. Right: Associated cumulative distribution function.

Consequently, for solving the original problem, (1) we solve the inversion problem directly and (2) we use the predicted value of the negotiable feature as mean of a normal distribution with the relative error as a relative standard deviation. We call this model *negotiable feature model*.

5 Prescription Problems with Negotiation

In this section, we propose a solution to solve the problems 5 to 8 that are described in Table 1. Our solution is based on the ideas we have presented about models that use negotiable features in the previous sections. The objective is to integrate the induced models into a negotiation process such that these models guide the negotiation. To do this, we first introduce the negotiation strategies we will use and, then, we describe the different scenarios that cover all cases. We are going to illustrate our proposals by using a general CRM problem of retailing (prescription applied to a plot selling scenario), where the (negotiable) input feature is the price (denoted by π) and the problem is a classification problem (buying or not).

In our problem, we know that customers have a "maximum price" per flat they are not meant to surpass. This maximum price is not known by the seller, but estimated with the data mining models. Conversely, the seller (real estate agent) has a "minimum price" (denoted by π_{min}) for each type of product, which typically includes the price the owner wants for the house plus the operational cost. This minimum price is not known by the buyer. Any increment over this minimum price is profitable for the seller. Conversely, selling under this value is not acceptable for the seller. Therefore, the seller will not sell the product if its price is under this minimum price that s/he knows. This means that the profit obtained by the product will be the difference

between the selling price and the minimum price: $Profit(\pi) = \pi - \pi_{min}$. Finally, in case that the maximum price is greater than the minimum price, there is a real chance of making a deal, and the objective for the seller is to maximise the expected profit, which is defined as follows:

$$E\_Profit(\pi) = \hat{p}(POS|\langle x_1, \ldots, x_{i-1}, \pi, x_{i+1}, \ldots, x_m\rangle) \cdot Profit(\pi) \quad (1)$$

where $\hat{p}$ is the estimated probability by the model.

5.1 Negotiation Strategies

The novel thing in this scenario is not only that we allow the seller to play or gauge the price to maximise the expected profit, but we also allow several bids or offers made to the same customer. This means that if an offer is rejected, the seller can offer again. The number of offers or bids which are allowed in an application is variable, but it is usually a small number, to prevail the buyer from getting tired of the bargaining.

We propose three simple negotiation strategies in this setting. For cases with one single bid, we introduce the strategy called "Maximum Expected Profit" (MEP). For cases with more bids (multi-bid) we present two strategies: "Best Local Expected Profit" (BLEP) strategy and "Maximum Global Optimisation" (MGO) strategy. Let us see all of them in detail below:

- Maximum Expected Profit (MEP) strategy (1 bid). This strategy is typically used in marketing when the seller can only make one offer to the customer. Each price for an instance gives a probability of buying. This strategy chooses the price that maximises the value of the expected profit. $\pi_{MEP} = argmax_\pi(E\_Profit(\pi))$. In Figure 2 right, the black dot is the MEP point (the maximum expected profit point). Note that, in this case, this price is between the minimum price (represented by the dashed line) and the maximum price (represented by the dotted line), which means that this offer would be accepted by the buyer.
- Best Local Expected Profit (BLEP) strategy (N bids). This strategy consists in applying the MEP strategy iteratively, when it is possible to make more that one offer to the buyer. The first offer is the MEP, and if the customer does not accept the offer, his/her curve of estimated probabilities is normalised taking into account the following: the probabilities of buying that are less than or equal to the probability of buying at this price will be set to 0; and the probabilities greater than the probability of buying at this price will be normalised between 0 and 1. The next offer will be calculated by applying the MEP strategy to the normalised probabilities. When the probability of buying which is associated to the price is the maximum probability (this is an infrequent situation) we cannot use the model any more, and the price will not be set to 0, because the expected profit would always be 0. Instead of this, the next offer is directly the half of the bid price. The pseudo-code is in Algorithm 1. Figure 3 left shows

the three probability curves obtained in three steps of the algorithm and Figure 3 right shows the corresponding expected profit curves. The solid black line on the left chart is the initial probability curve and the point labeled by 1 on the right chart is its MEP point. In this example, the offer is rejected by the customer (this offer is greater than the maximum price), so probabilities are normalised following the process explained above. This gives a new probability curve represented on the left chart as a dashed red line and its associated expected profit curve (also represented by dashed red line on the chart on the right), with point 2 being the new MEP point for this second iteration. Again, the offer is not accepted and the normalisation process is applied (dotted green lines in both charts). In order to illustrate the case where the normalisation plummets the probabilities too, Figure 4 shows the BLEP strategy when the probability associated to the MEP point in each iteration is the maximum probability.

– Maximum Global Optimisation (MGO) strategy (N bids). The objective of this strategy is to obtain the N offers that maximise the expected profit:

$$\begin{aligned}\pi_{MGO} &= argmax_{\langle\pi_1,\ldots,\pi_N\rangle}(E\_Profit(\langle\pi_1,\ldots,\pi_N\rangle)\\ &= argmax_{\langle\pi_1,\ldots,\pi_N\rangle}(\hat{p}(POS|\pi_1)\cdot Profit(\pi_1)\\ &\quad +(1-\hat{p}(POS|\pi_1))\cdot\hat{p}(POS|\pi_2)\cdot Profit(\pi_2)+\ldots\\ &\quad +(1-\hat{p}(POS|\pi_1))\cdot\ldots\cdot(1-\hat{p}(POS|\pi_{N-1}))\cdot\\ &\quad \hat{p}(POS|\pi_N)\cdot Profit(\pi_N))\end{aligned}$$

The rational of the previous formula is that we use a probabilistic accounting of what happens when we fail or not with the bid. Consequently, optimising the previous formula is a global approach to the problem.
Computing the N bids from the previous formula is not direct but can be done in several ways. One option is just using a Montecarlo approach [19] with a sufficient number of tuples to get the values for the prices that maximise the expected profit. Figure 5 right shows the three points given by the MGO strategy for the probability curve on Figure 5 left. As we can see, the three points are sorted in decreasing order of price.

For the three previous strategies, it is clear that they will only work if the data mining models perform relatively accurate predictions in terms of probability estimation[3].

Next, we will investigate the application of the previous three methods to the last four cases in Table 1.

5.2 Scenario with one Product and one Customer

We start with the simplest negotiation scenario, where there are only one seller and one buyer who both negotiate for one product. The buyer is interested in

[3] In the last two strategies, we do not consider whether the offer is below the seller's minimum price. Strictly, this is not part of the strategy but it is rather more related to ascertain which of cases 5, 6, 7 or 8 we are dealing with, and also with the issue of whether we have other customers and we prefer to stop offering the product that to get closer to the minimum price.

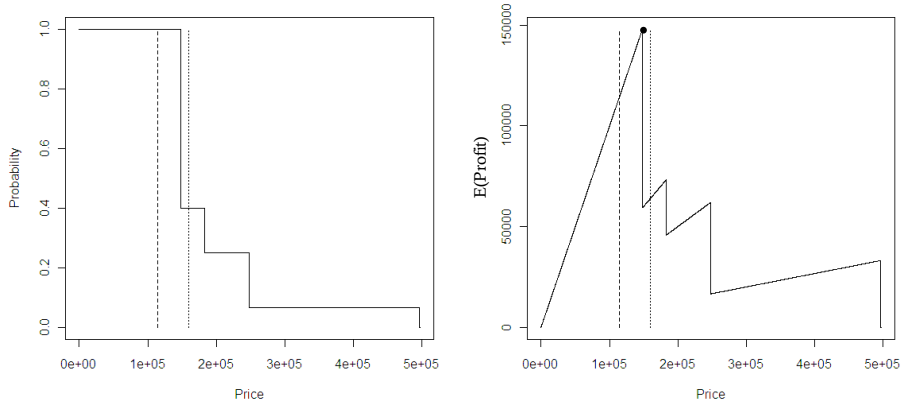

Fig. 2 Example of the MEP strategy. **Left**: Estimated probability. **Right**: Associated expected profit.

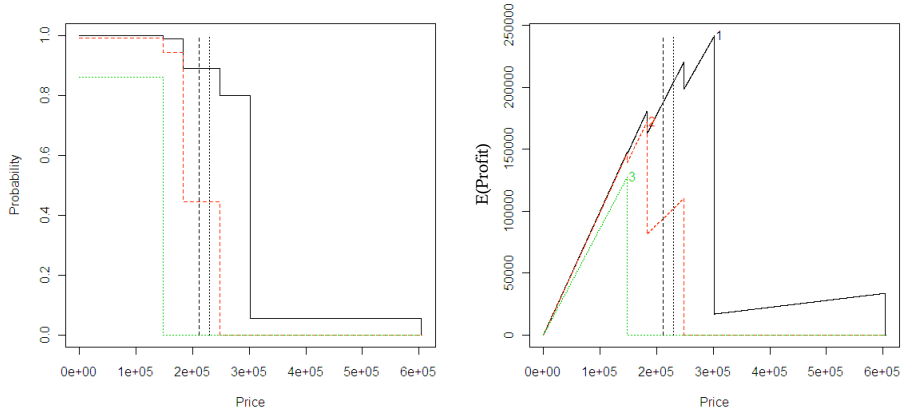

Fig. 3 Example of the BLEP strategy. **Left**: Estimated probability. **Right**: Associated expected profit.

one specific product. S/he likes the product and s/he will buy the product if its price is under a certain price that s/he is willing to pay for this product. It is clear that in this case the price holds the conditions to be a negotiable feature. It is sensitive, since if we reduce price to 0, the probability of having class POS approaches 1 and if we increase price to a very large amount, the probability of having class NEG approaches 1. And, finally, it is monotonic, since the relation between price and the class order $NEG \prec POS$ is monotonically decreasing. Since product and customer are fixed, we only have one degree of freedom: the price.

Obviously, the goal of the seller is to sell the product at the maximum possible price (denoted by π_{max}) which is defined as the value such that both the following equalities hold:

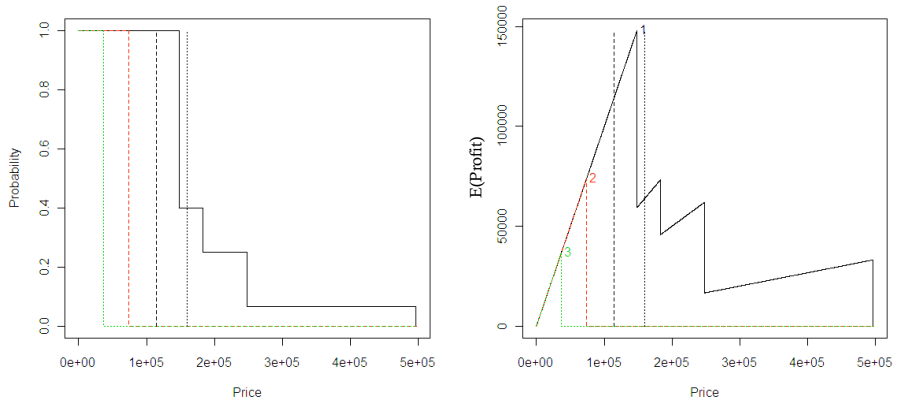

Fig. 4 Example of the BLEP strategy. **Left**: Estimated probability. **Right**: Associated expected profit.

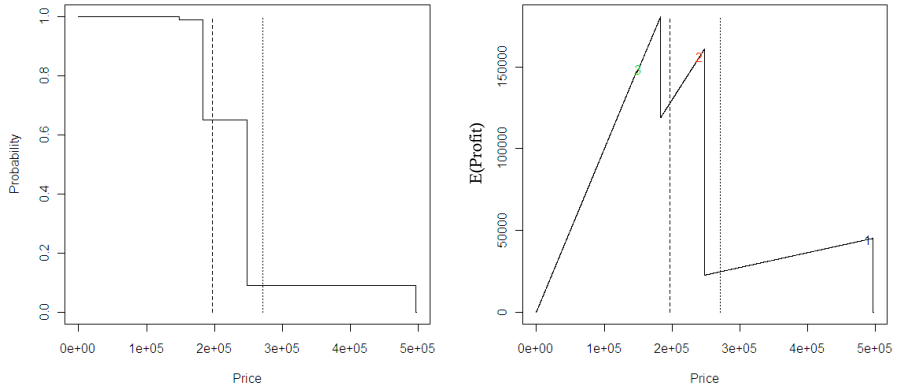

Fig. 5 Example of the MGO strategy. **Left**: Estimated probability. **Right**: Associated expected profit.

$$f(x_1, \ldots, x_{i-1}, \pi_{max}, x_{i+1}, \ldots, x_m) = POS$$
$$f(x_1, \ldots, x_{i-1}, \pi_{max} + \epsilon, x_{i+1}, \ldots, x_m) = NEG, \forall \epsilon > 0.$$

In other words, the use for the model is: "Which is the maximum price at which I can sell this product to this customer?" Logically, the higher the price the lower the probability. So the goal, as we said at the beginning of Section 5, is to maximise the expected profit calculated by formula (1) where $\hat{p}$ is the estimated probability given by the negotiable feature model.

To ease notation we will denote $\hat{p}(POS|\langle x_1, \ldots, x_{i-1}, \pi, x_{i+1}, \ldots, x_m\rangle)$ as $\hat{p}(POS|\pi)$. Consequently, we can express formula (1) as:

$$E(Profit(\pi)) = \hat{p}(POS|\pi) \cdot Profit(\pi),$$

with the additional constraint, as mentioned, that $\pi \geq \pi_{min}$.

Algorithm 1: BLEP strategy

Require: N, epf (estimated probability function or curve)
Ensure: π_{BLEP}
  $\forall x, epf(x) \leftarrow \hat{p}(POS|x)$
  $\pi_1 \leftarrow \pi_{MEP}$
  $\pi \leftarrow \pi_1$
  for $\pi_i, i \in 2..N$ **do**
    if $epf(\pi) \neq \max_{x \in 0..\infty}(epf(x))$ **then**
      $\forall x, epf(x) \leftarrow 0$
      if $epf(x) \leqslant epf(\pi)$ **then**
        $epf \leftarrow normalise(epf, epf(\pi), \max_{x \in 0..\infty} epf(x))$
        $\{normalise(f(x), mix, max)$: returns normalised function of $f(x)$ from values min and max to $[0..1]\}$
      end if
      $\pi_i \leftarrow \pi_{MEP}$
      $\pi \leftarrow \pi_i$
    else
      $\pi_i \leftarrow \pi \div 2$
      $\pi \leftarrow \pi_i$
    end if
  end for
  $\pi_{BLEP} \leftarrow \langle \pi_1, \ldots, \pi_N \rangle$

So, if we have a model which can estimate probabilities for the positive class, we can use the previous formula for the expected profit to choose the price that has to be offered to the customer. If probabilities are well estimated, for all the range of possible prices, this must be the optimal strategy if there is only one bid. In Figure 6 we show an example of the plots that are obtained for the estimated probabilities and expected profit.

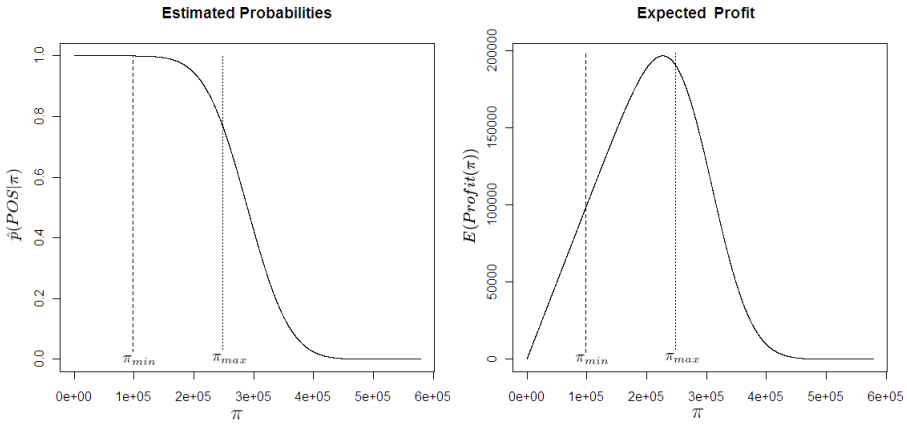

Fig. 6 Left: Example of estimated probabilities. Right: Associated expected profit. The minimum and maximum price are also shown.

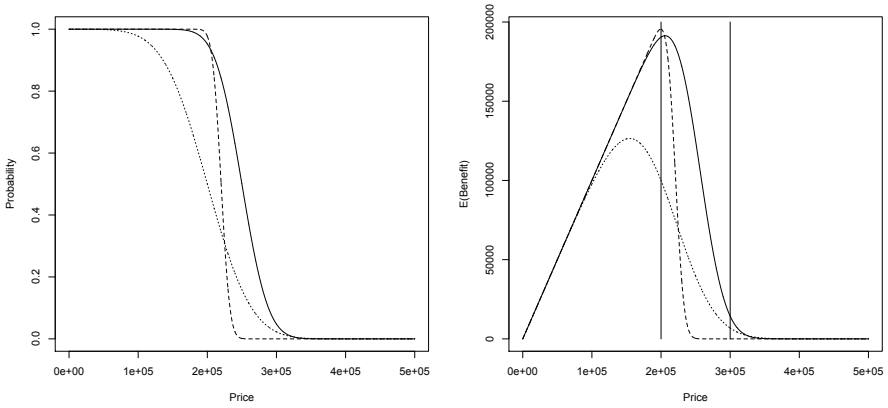

Fig. 7 Probabilistic buying models of 3 different customers approximated by 3 normal distributions with $\mu_1 = 250,000$ and $\sigma_1 = 30,000$, $\mu_2 = 220,000$ and $\sigma_2 = 10,000$, and $\mu_3 = 200,000$ and $\sigma_3 = 50,000$. **Left**: Probability distribution function. **Right**: Associated expected profit.

5.3 Scenario with several Products and/or several Customers

In this section we are going to study the cases 6, 7 and 8 in Table 1. The cases 6 and 7 correspond to the cases with more than one product or more than one customer, while in the case 8 there are several products and several customers. As we will see, they can be understood as an extension of case 5 combined with the rankings of customers and products that are used in cases 2 and 3 in Table 1.

In case 6 in Table 1 (one kind of product, negotiable price, and C customers), there is a curve for each customer (Figure 7, Left), which are similar to the curve in case 5. If the seller can only make one offer to the customers, the seller will offer the product at the price that gives the maximum expected profit (in relation to all the expected profit curves) to the customer whose curve achieves the maximum. However, if the seller can make several offers, the seller will distribute the offers along the curves following a negotiation strategy. In this case, the seller not only changes the price of the product, but the seller can also change the customer that s/he is negotiating with, depending on the price of the product (that is, by selecting the customer in each bid who gives the greatest expected profit at this price). Therefore, these curves can be seen as a ranking of customers for each price.

Case 7 in Table 1 (N kind of products, a negotiable price, and one customer) is symmetric to case 6. Instead of one curve for each customer, there is one curve for each product. In this case, the curves represent a ranking of products for that customer. The learned data-mining models will help the seller to make the best decision about which product the seller offers to the

customer and at what price. Figure 7 is also an example of this case since the curves would represent three different products to be offered to one customer.

Case 8 in Table 1 (N kind of products, a negotiable price, and C customers) is the most complete of all.

The objective is to offer the products to the customers at the best price in order to obtain the maximum profit. Multiple scenarios can be proposed for this situation: each customer can buy only one product; each customer can buy several products; if the customer buys something, it will be more difficult to buy another product; there is limited stock; etc. But if we understood it as the other two, the way in which it is solved does not differ to cases 6 and 7.

In cases 6, 7 and 8, we typically work with only one data mining model which has the customer's features and the product's features (one of them being the negotiable feature) as inputs. We can, of course, define C different models in case 6, N different models in case 7, or even C, N or CxN different models for case 8. Nonetheless, this is not necessary and the higher the number of models is the more difficult is to learn and use them and is prone to overfitting.

As a result, to solve cases 6, 7 and 8, we propose extending the classical concept of ranking customers or products to expected profit curves in order to obtain a ranking of customers or products for each price (similar to cases 2 and 3). For example, Figure 7 shows that, for a price of 300,000 euros the most appropriate customer is the one represented by the solid line, the second one is the customer represented by the dotted line, and the least appropriate one is represented by the dashed line. The situation changes for a price of 200,000 euros; at that point the most appropriate customer is the one represented by the dashed line, the second one is the customer represented by the solid line, and the least one is the one represented by the dotted line. Therefore, an important property of these probabilistic buying models is that there is a change in the ranking at the point where two curves cross.

Graphically, the most appropriate customer or product (the top of the ranking) for each price is represented by the envelope curve. Therefore, in the cases 6, 7 and 8 there are several curves, but the envelope curve must be calculated having, as a result, only one curve. Consequently, we can apply the same negotiation strategies applied to the case 5 to the envelope curve of cases 6, 7 and 8.

Example 3 We are going to explain the negotiation strategy that the seller will follow by means of an example of the case 6 (one kind of product, a negotiable price, and C customers), because the process will be the same for the cases 7 and 8, but with more curves. Therefore, we start with the simplest situation with two customers and one product.

In Figure 8, there are two curves representing the buying probabilities of two different customers. The buying probability of the first customer follows a normal distribution with $\mu_1 = 400$ and $\sigma_1 = 100$, and it is represented by a dashed line. The buying probability of the second customer follows a normal distribution with $\mu_2 = 300$ and $\sigma_2 = 200$, and it is represented by a dotted

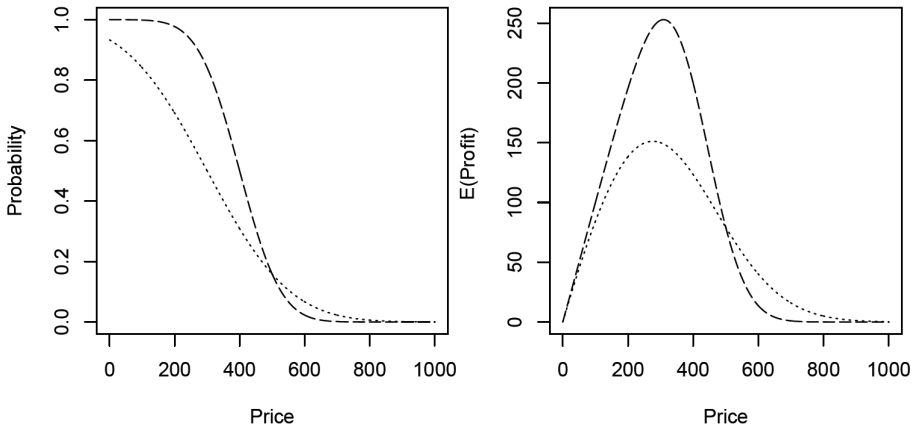

Fig. 8 Probabilistic buying models of 2 different customers approximated by 2 normal distributions with $\mu_1 = 400$ and $\sigma_1 = 100$ (dashed line), and $\mu_2 = 300$ and $\sigma_2 = 200$ (dotted line). **Left**: Probability distribution function. **Right**: Associated expected profit.

Table 2 **Left:**Trace of the negotiation process. **Right:**Trace of the negotiation process with the ordering pre-process.

Offer	Price	Customer	Accepted
1	309	1	No
2	214	1	No
3	276	2	No
4	149	1	No
5	101	1	No
6	150	2	Yes

Offer	Price	Customer	Accepted
1	309	1	No
2	276	2	No
3	214	1	No
4	150	2	Yes

line. These are the probabilities; however, the actual values (unknown by the seller) is that the maximum buying price for customer 1 is 100 euros and 150 euros for customer 2.

We assume a simple negotiation process for this example. The negotiation strategy that we use is the Best Local Expected Profit (BLEP) strategy explained in section 5.1. The trace of the negotiation process is described in Table 2 (Left) and shown graphically in Figures 9 and 10. In each iteration, the maximum of the functions is calculated (the envelope curve). The envelope curve is represented by a solid line in Figures 9 and 10.

Note that as Table 2 (Left) shows, the third offer is greater than the second one. This is because there is more than one customer in the negotiation process and the offer is made at the price that maximises the expected profit at each iteration. Therefore, it is easy to propose an improvement for this negotiation strategy with a limited number of offers, which is similar to BLEP with n bids. This improvement is a pre-process that consists in calculating the n points and ordering them by the price before starting the negotiation. Following the example shown in Table 2 (Left), if there are only 4 bids no one will buy the product. However, with this improvement (the pre-process) customer 2 will buy the product at a price of 150 euros as shown in Table 2 (Right).

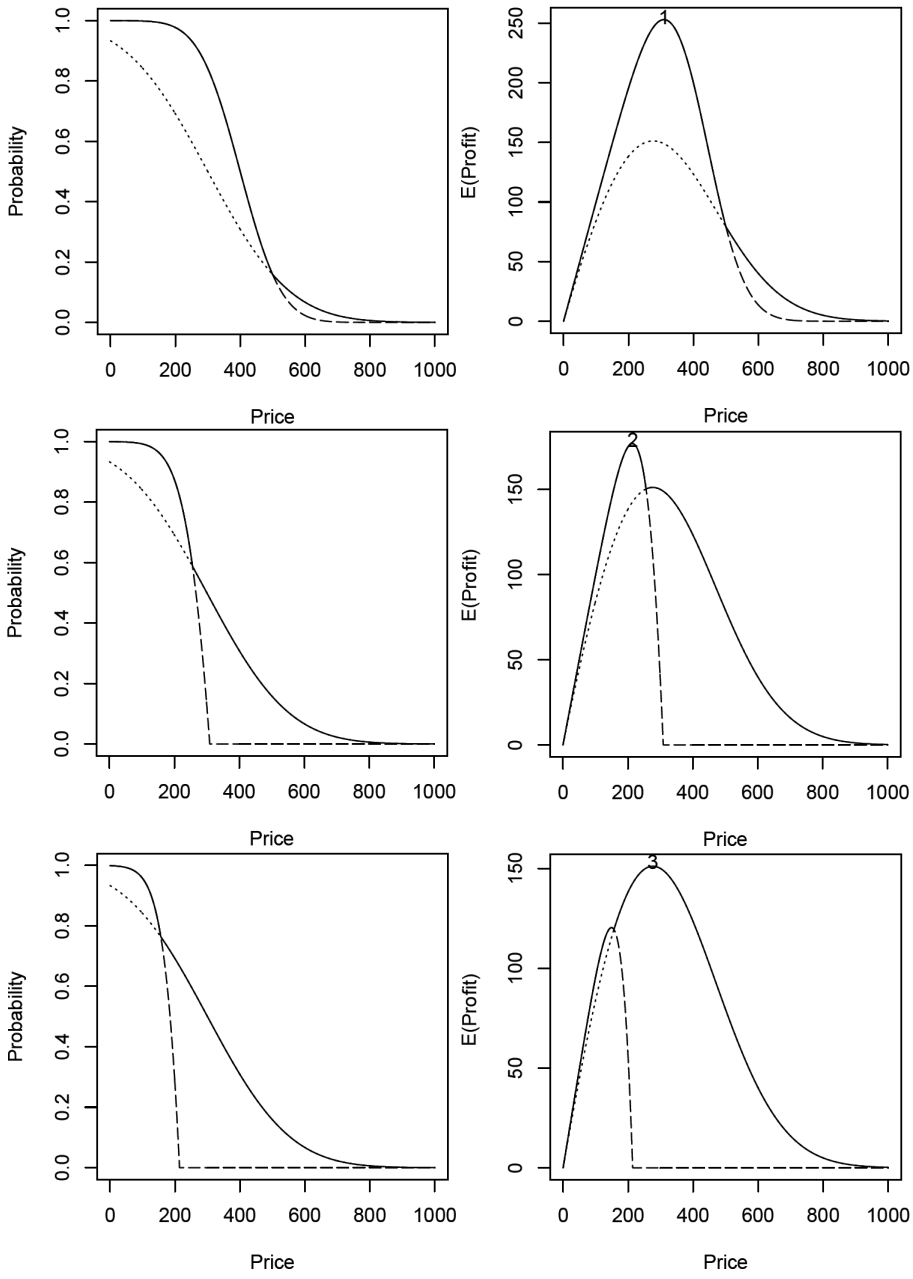

Fig. 9 Points 1, 2 and 3 in the negotiation process. **Left**: Probability distribution function. **Right**: Associated expected profit.

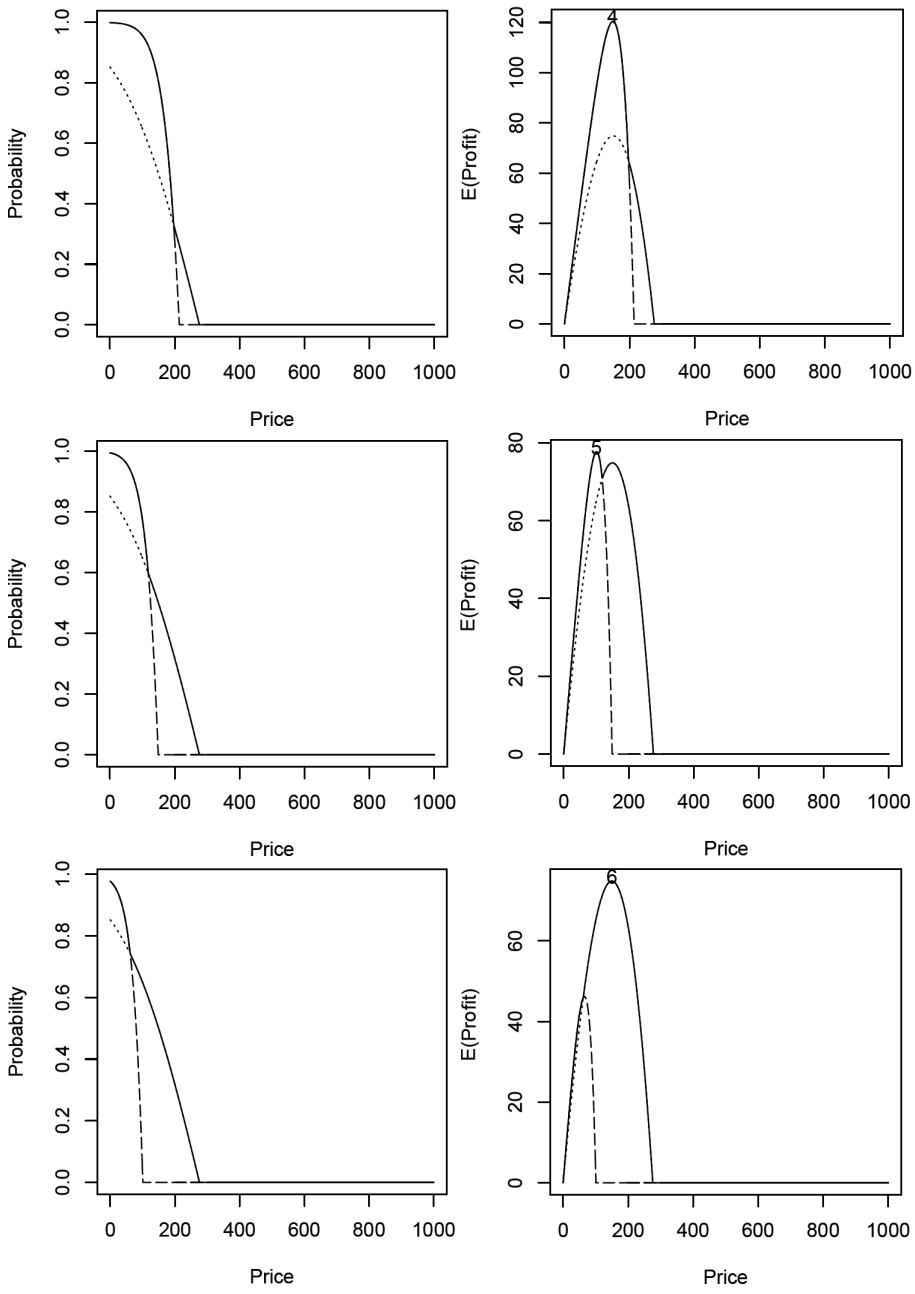

Fig. 10 Points 4, 5 and 6 in the negotiation process. **Left**: Probability distribution function. **Right**: Associated expected profit.

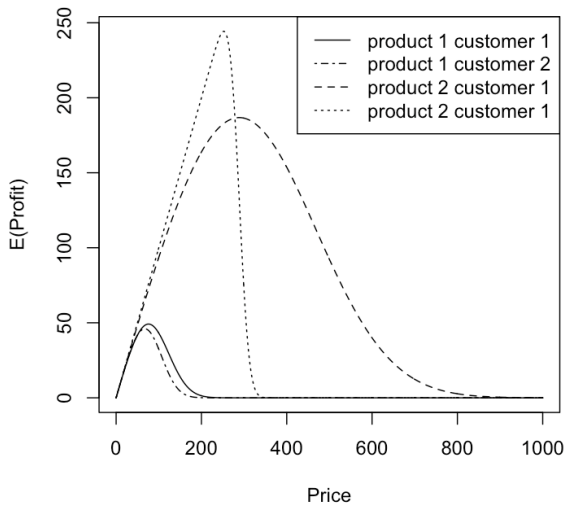

Fig. 11 Example of probabilistic models of two customers and two products.

This negotiation scenario suggests that other negotiation strategies can be proposed for application to problems of this type in order to obtain the maximum profit. One problem with the BLEP strategy is that it is very conservative. It might be interesting to implement more aggressive strategies that make offers at higher prices (graphically, more to the right). A negotiation strategy that attempts to do this is the Maximum Global Optimisation (MGO) strategy (with n bids). The objective of this strategy is to obtain the n offers that maximise the expected profit by generalising an optimisation formula that was presented in section 5.1.

In case 6 (One kind of product, a negotiable price, and C customers), we have presented an example with two customers and one product, but it would be the same for more than two customers. In the end, there would be one curve for each customer or product, and the same negotiation strategies could be applied.

Case 7 (N kind of products, a negotiable price, and one customer) is the same as case 6, but the curves represent the buying model of each product for each customer, and a ranking of products will be obtained for each price.

Case 8 (N kind of products, a negotiable price, and C customers) can be studied using the same concept of expected profit curves, but there will be $N \times C$ curves. For each of the N kind of products, there will be C curves that belong to the buying model of each customer. Figure 11 presents a simple example with two products and two customers. Several settings could be possible: each customer can only buy one product, there is limited stock, etc. Therefore, the curves will change or disappear in real time, depending on the setting of the

problem. In this work we are always offering the best product to the best customer, but there is no problem in offering more than one product to the same customer or to multiple customers[4]. If we only have one product, the first customer in answering will obtain the product. For example in the case of offering the two topmost ranked products to a customer, we would work as follows. First, we would obtain the most desirable product, in the same way as case 7 in table 1 (N kinds of products, a negotiable price, and one customer). Second, the curve of this product would be deleted. And third, the most desirable of the remaining products would be obtained, again in the same way as case 7.

6 Experiments

Experiments have been performed by using real data collected from an estate agent[5]. We have information of 2,800 properties (flats and houses) that were sold during 2009 in the city of Valencia (Spain), for which we have the following attributes ("district", "number of rooms", "square metres", "owner's price" and "selling price"). "Owner's price" is the price which the owner wants to obtain for the property. "Selling price" is the final price at which the purchase took place.

We assume that the "selling price" is some kind of "market price", which is usually closer to the "maximum price". Although this is not always true because in some cases the buyer could have paid more than this for the property, it is generally a good approximation as we discussed in Section 1. In any case, it is almost impossible to obtain the real "maximum price", because it is very difficult that a customer says the maximum price that s/he could pay for a product.

We have randomly split the dataset, using a uniform distribution without replacement, into a training set and a test set. 10% of the data are for training and the rest to test. This tries to simulate a realistic situation when there are not too many data for training. Therefore, the results refer to 2,520 properties, and learning is made from 280 flats. We applied the solutions proposed in Section 4 to the data, namely the idea of populating the dataset to learn a better classification model and the idea of using regression models (inverted problem). In particular we have used the "improved classifier" solution (presented in section 4.2. In particular we learnt a $J48$ (WEKA implementation of $C4.5$ algorithm [26]) decision tree classifier (with Laplace correction and without pruning), implemented in the data mining suite WEKA [31]. It has been learned using example generation (10 positive examples and 10 negative examples for each original example using the negotiable feature rules explained in Section 4.2, so condition 2 holds for all the instances and $\tau = 1$). Since the predicted probability curve given by a classifier (such as the

[4] Note that we do not need a multinomial model for this. We only need to determine when two offers are close in the ranking.

[5] Data can be found at: "http://tinyurl.com/2usbdfj" in WEKA format.

J48 classifier) typically shows discontinuities and strong steps when varying a negotiable feature, we have smoothed the curve with a low-pass filter with Bartlett overlapping window [30]. The parameter of the window has been set to the "minimum price" divided by 4. This value were set after making some experiments (divided by 2, 3, 4 and 5) with some flats and observing that this value smoothed the curves properly. In Figure 12 we can observe the difference between the original curve and the smoothed curve. The "inverting problem presentation" solution (presented in Section 4.3) has been implemented with the *LinearRegression* and $M5P$ [25] regression techniques (both with their default parameters), also from WEKA[6].

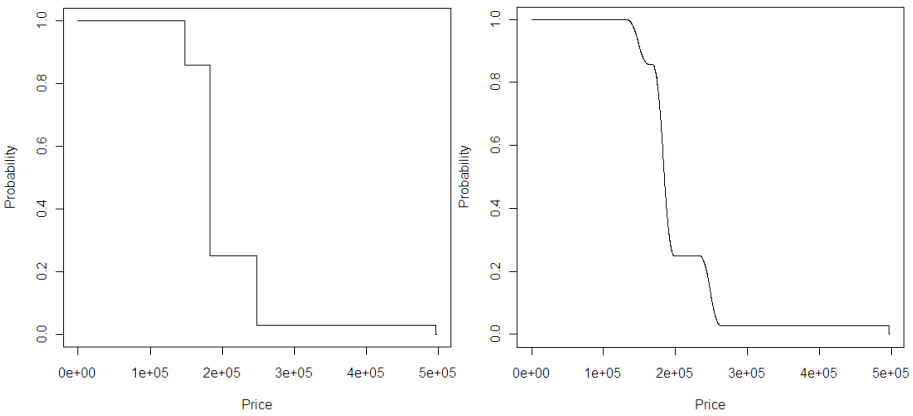

Fig. 12 Left: Example of estimated probabilities. Right: Estimated probabilities which have been smoothed by a low-pass filter with Bartlett overlapping window.

These three learning techniques have been used to guide the three negotiation strategies explained in section 5.1. For the MGO strategy we have used a Montecarlo approach [19]: 3 prices are selected using a uniform random distribution from the range of prices and the value for these 3 points is calculated using the formula in section 5.1, this operation is repeated 1,000 times and the triplet that obtain the maximum value for the formula is chosen.

In Table 3 we can observe the results for case 5 in Table 1 (one kind of product, a negotiable price, and one customer), obtained for each method, in terms of number of sold properties, deal price (in euros) and profit (in euros). In Table 4 we show the results for case 7 in Table 1 (N kinds of

[6] The source code of the algorithm which computes the probability can be obtained at: "http://tinyurl.com/3xxpxyp". The version of the Java Development Kit used is "jdk1.6.0_14" that can be downloaded at: "http://www.sun.com/java/" and the learning algorithms come from the Weka API (version 3.6.1) "http://www.cs.waikato.ac.nz/ml/weka/". The negotiation strategies have been implemented using version 2.9.2 of R (R-project statistical package) "http://www.r-project.org/" and the script with these algorithms is accessible at: "http://tinyurl.com/33e5up6".

products, a negotiable price, and one customer). Concretely, we have set the number of different products to five, i.e., $N = 5$. For these experiments with 1 customer and 5 flats, we chose each group of 5 flats randomly using a uniform distribution and without replacement. This means that we have grouped the 2,520 flats in groups of 5 flats, having 504 groups of 5 flats. Each group of 5 flats is offered to the customer and s/he can only buy one of the flats. As we have explained in section 5.3, the envelope curve of the 5 curves is calculated and the negotiation strategies are applied to it.

In Table 3 and Table 4 we compare the MEP, BLEP and MGO strategies with these baseline methods:

- Baseline method (1 bid or N bids) (1 customer and 1 product). One of the simplest methods to price a product is to add a margin (a percentage) to its minimum price (or base cost). Instead of setting a fix percentage arbitrarily, we obtain the percentage (called α) such that it obtains the best result for the training set. For example, in our experiments, the best α is 0.8, so it is expected that the best profit will be obtained by increasing the minimum price of the properties in an 80%. If we have only 1 bid, we will increase the minimum price of the flat by α. But, if we have N bids, we will have one half of the bids with a value of α less than the calculated α and the other half of the bids with a value of α greater than the calculated α. In particular, the value of α will increase or decrease by $\alpha/(N + 1)$ in each bid. For example, in our experiments for 3 bids, the three values of α for three bids would be 100%, 80% and 60%. Therefore, the first offer would be an increase of 100% over the minimum price of the product, the second an increase of 80% and the third an increase of 60%.
- Baseline method (the most expensive/the cheapest) (1 customer and M products). When the seller has several products to offer to the customer, it is not as clear as the previous case how to create a baseline method, because there are several products with different prices. We have proposed two baseline methods: one associated with the most expensive product of the M products, and the other associated with the cheapest product of the M products. In other words, the baseline methods in these cases are the same as the baseline method (1 bid or N bids) (1 customer and 1 product), but in these cases the increased price is the price of the most expensive product or the price of the cheapest product.

In Table 3 we also show the results of two reference methods. The methods "All flats sold at π_{min}" and "All flats sold at π_{max}" show the hypothetic cases when all the flats are sold at the minimum price or at the maximum price. In Table 4 the methods "Selling the most expensive" and "Selling the cheapest" show the hypothetic cases when, in all the groups of 5 flats, the most expensive flat of the 5 have been sold to its maximum price, or the cheapest flat of the 5 have been sold to its maximum price.

In order to analyse the results, let us first focus on Table 3, which shows results for one customer and one flat. For one possible bid, we have that all the methods based on data mining get better results. The solution using an

1 customer and 1 flat			
Method	**Sold flats**	**Deal price**	**Profit**
Reference			
All flats sold at π_{min}	2,520	534,769,904	0
All flats sold at π_{max}	2,520	712,580,216	177,810,312
1 bid			
Baseline (80%)	290	74,664,508	33,184,226
MEP ($J48$)	1,094	244,102,200	42,034,896
MEP ($LinearRegression$)	1,681	341,147,000	38,372,983
MEP ($M5P$)	1,707	342,805,800	38,580,279
3 bids			
Baseline (100%, 80%, 60%)	635	165,443,695	67,226,421
BLEP ($J48$)	1,211	260,298,700	43,474,928
BLEP ($LinearRegression$)	1,746	352,112,700	39,574,602
BLEP ($M5P$)	1,769	353,106,100	39,698,991
MGO ($J48$)	1,700	422,209,800	84,028,502
MGO ($LinearRegression$)	1,918	477,247,900	95,323,551
MGO ($M5P$)	1,939	487,771,600	98,376,548

Table 3 Results obtained for one product and one customer by the negotiation strategies, baseline methods and reference methods (minimum and maximum profit). Deal price and profit measured in euros.

1 customer and 5 flats			
Method	**Sold flats**	**Deal price**	**Profit**
Reference			
Selling the most expensive	504	233,943,288	58,092,977
Selling the cheapest	504	85,385,709	22,641,094
1 bid			
Baseline (80%) (the most expensive)	48	20,822,533	9,254,459
Baseline (80%) (the cheapest)	74	11,763,632	5,228,281
MEP ($J48$)	180	74,025,300	13,043,111
MEP ($LinearRegression$)	242	79,143,500	6,506,317
MEP ($M5P$)	226	72,390,900	5,613,714
3 bids			
Baseline (100%, 80%, 60%) (the most expensive)	123	50,497,241	18,936,465
Baseline (100%, 80%, 60%) (the cheapest)	144	22,025,593	8,259,598
BLEP ($J48$)	354	116,478,000	20,468,778
BLEP ($LinearRegression$)	434	123,971,100	11,146,363
BLEP ($M5P$)	431	121,216,400	10,491,057
MGO ($J48$)	288	115,919,700	20,213,901
MGO ($LinearRegression$)	339	142,570,700	24,541,886
MGO ($M5P$)	344	147,656,200	25,109,410

Table 4 Results obtained for one customer and 5 products by the negotiation strategies, baseline methods and reference methods. Deal price and profit measured in euros.

extended dataset but preserving the original task (J48 classifier) is slightly better than the problem inversion methods (Linear Regression and M5P regressor). Taking a look at the number of flats sold, it suggests that MEP with regression somehow underestimates the ideal price in this situation. For three bids, the picture changes. The baseline method is now better than BLEP. This is expected since BLEP just chooses the local optimum each time and disregards the overall picture. On the contrary, the MGO clearly surpasses the other methods, which shows that a global formulation is necessary to solve the case for several bids. If we take a look at the method, regression (and M5P in

particular) is the best method for this case. As a wrapping-up of the results for one customer and one flat we can say that for one bid, MEP with a J48 classifier gives the best results, while the MGO with the M5P regressor is the best combination.

Now let us focus on Table 4, which shows results for one customer and 5 flats. For one possible bid, we have that all the methods based on data mining get better results than the "baseline (80%) (the cheapest)" but not for the "baseline (80%) (the most expensive)". Only the solution using an extended dataset but preserving the original task (J48 classifier) is better than both baselines. Again, taking a look at the number of flats sold, it suggests that MEP with regression somehow underestimates the ideal price in this situation. For three bids, the picture changes again. The "baseline method the cheapest" is the worst method while "the baseline method the most expensive" is now better than BLEP using regression, and comparable to BLEP using classification. This is again expected since BLEP just chooses the local optimum each time and disregards the overall picture. On the contrary, MGO clearly surpasses the other methods (with the only exception that MGO with J48 is worse than BLEP with J48). This also shows that a global formulation is necessary to solve the case for several bids. If we take a look at the method, regression (and M5P in particular) is the best method for this case. As a wrapping-up of the results for one customer and 5 flats we can say that for one bid, MEP with a J48 classifier gives the best results, while MGO with the M5P regressor is the best combination.

Consequently, the results for one customer and one flat are in agreement with one customer and five flats (case 5 in Table 1). Since the problem with C customers and one flat (case 6 in Table 1) is symmetrical wrt. one customer and N flats (case 7 in Table 1), similar results are expected for the case of C customers and N products (case 5 in Table 1), since its solution is similar to the other two cases.

In conclusion, the MEP or BLEP negotiation strategies can obtain good results, but the MGO method is more robust, because it is a global optimisation method.

7 Related Work

This work incorporates ideas from data mining [14] (machine learning and statistics), marketing research [7], negotiation [16], agent theory [27] and, decision theory [21], in order to address a taxonomy of prescription problems with basic negotiation issues in the area of marketing and customer-relationship management. In this section we describe the possible connections between our work and some previous or related approaches. One goal of this section is to clearly see where we differ from other approaches. The difference may be found because we address different (but related) problems or because we use different techniques (and hence a comparison should be made). In both cases, however, we can take some ideas for extending our approach to more complex

or general problems or to address the same problem with better techniques. Let us see this.

The first field we come up is decision theory. In prescription problems, we are always concerned about what to sell and to whom. If we have a utility function, and we have a probabilistic model for a set of decisions (choosing the product or choosing the customer), we can derive an expected utility for each set of decisions, and try to maximise this. If there is no interaction at all, we have a list of expected utilities, from which we can construct a rank and, design, for instance, a mailing campaign. If there is a finite sequence of interactions, we can analyse that with a probabilistic decision tree or with transition graphs. If this interaction may get larger or infinite, then we may require a Markov Decision Process (MDP) [24], assuming that the Markov property holds. In this case, many techniques from dynamic programming [5] and reinforcement learning [28] can be used. Although our MGO method may resemble some of the methods in reinforcement learning, we have used a Montecarlo approach, because we have an infinite multidimensional set of inputs, and we want the probabilistic model to be treated as a black box (it is a data mining model which can be linear, non-linear, non-continuous). We could also study the application of linear programming techniques for this problem.

A prescription problem with negotiable attributes not only depends on what to sell and to whom. It also deals with features, such as prices. And these features are typically continuous, so we cannot address this with discrete approaches such as MDP, which are discrete-time and discrete-action decision processes, unless we discretise these attributes in a reduced set of bins and we augment the MDP framework to consider an order relation between the actions (as the order relation we have defined for our inputs, and outputs). Although discrete-time is not a problem here, we need to look to continuous-action decision processes, as in control theory [17]. However, we fail to identify a good representation of the problem here in such a way that we do not recognise the conditions which are typically required in control problems. Additionally, we have few feedback interactions from the customers (and very few if we only consider one customer at a time), we do not have many continuous outputs from the system (just a purchase or not, not even a degree of satisfaction), so the idea of gradually adjusting and gauging which is typical in process control does not hold here either.

The closest approaches are from the broader areas of game theory [12], negotiation policies [16] and multi-agent optimisation [32]. However, much of the results and techniques have been specialised to very specific cases (see, e.g. [29]), while there are only a few general principles from game theory, such as the Nash equilibrium if the agent competes (as in our case). However, we are not concerned about the global optimum, but the optimum for one agent (typically the seller), and there are many games held at the same time sharing some issues (products and customers).

Back to the field of CRM and marketing, it seems that cross-selling, up-selling and recommender systems may be useful here, as we have already mentioned in the introduction. There are some approaches that employ data

mining techniques to configure cross-selling campaigns. For instance [18] applies multinomial logit on a CRM cross-sell application. Another example is [23]. This work presents a structural multivariate probit model to investigate how customer demand for multiple products evolves over time. However, it is not straightforward to apply this kind of data mining techniques in our framework. The issue is the family of problems we are addressing; we are not thinking about a long-term customer relationship. The examples we are using are typically related to selling houses, cars, or other products which are not repeatedly purchased by the same customer. In fact, it is more likely (custom and acceptable) to negotiate and use different prices on these products for each customer than to negotiate or use different prices for a bar of chocolate. At least at this moment of the taxonomy and the kinds of problems we consider, there is no follow-up about the customer, no real concern about customer's churn, etc. The main goal is to maximise profits for a buyer that will not been seen again after a purchase (unless a complaint or a refund).

There is an interesting relation between the notion of negotiable feature and Choice Based Conjoint (CBC) Analysis ([13]). The main objective of Conjoint Analysis is to measure an individual's or a population's preference on a set of parameters and levels. CBC Analysis is a special family of Conjoint Analysis techniques that have to choose between a set of options for several parameters. In the context of market research, subjects are customers and parameters are product's features. The output of the process is an ordinal scale which ranks all the options or, more frequently, a scale in which every option is given a value. In other words, Conjoint Analysis allows for a proper ranking of the features. Conjoint analysis presents the problem that one option cannot compensate the bad value of other options (e.g. we will not buy a flat at any price if it does not have a lift). This is related to the parameter τ in our definition of negotiable feature. An option (or feature) can be made irrelevant given the values of other features. Adaptive Choice Based Conjoint (ACBC) Analysis is an extension of Choice Based Analysis which allows for non-compensatory decision rules as in the previous example. It is frequent to see a first ACBC analysis to define which features are relevant, and second, to apply a CBC to rank them. This would be appropriate in cases where we do not exactly know the attributes which are negotiable. However, we have to clarify that CBC is based on examples which are preference expressions (e.g. 'I prefer flat A with feature values a_1, a_2, ..., a_n, over flat B with feature values b_1, b_2, ..., b_n'). In our case, our examples are like 'Customer bought (or didn't) flat A with feature values a_1, a_2, ..., a_n'. Even in cases where the data does not come from choices (general Conjoint Analysis) there is typically a pre-designed research survey, with preference questionnaires that may take several minutes. This scenario is far from the assumptions we are making here about a historical datasets with actual transactions, instead of a survey to gather information. In our case, we would prefer to adapt a classical feature selection method instead. Nonetheless, in cases where the survey can be performed, CBC analysis can be good tool to determine the negotiable attributes, especially in cases where we want to optimise the offer for more than one negotiable feature at a time,

because the ranking of feature relevance as well as their range of acceptable values can help in the combinatorial problem of finding the best set of values for the first and subsequent offers.

8 Conclusions

In this paper, we have investigated a new family of prescription problems using data mining models, where one or more features are negotiable. These problems have motivated the extension of the taxonomy of prescription problems, and the development of a series of techniques to solve the optimisation problem of maximising the result that can be obtained with the models. A key notion has been the handling of the inversion problem which appears when transforming an input (negotiable) feature into an output feature, which can turn a classification problem into a regression problem and viceversa. The probabilistic estimation for the new output feature has been solved for both cases (probabilistic classification and regression), so producing probability estimation curves and profit curves. Using these curves we have been able to devise several negotiation strategies, which have been proven to behave better as long as a more global view is taken, which usually implies more complex maximisation problems which, due to characteristics of the data mining model, have been addressed with a Montecarlo approach.

The scenario we have used as a guiding thread for the paper shows a realistic problem in the CRM field, where data mining can help a seller to make a good decision about which product should be offered to which customer and at what price, in order to obtain as much overall profit as possible. However, the techniques presented here are applicable to many other prescription problems, inside the CRM field or outside (e.g. medicine, education, law, ...), or many other situations where some kind of negotiation takes place using data mining models.

This work has therefore been focused on the model deployment stage, which is becoming a scientifical problem itself, with much more entity and shape than some years ago, when data integration and processing, data modelling and evaluation were the data mining stages where the main computational effort and techniques were developed. Data deployment in a context of global maximisation requires the hybridisation of techniques from several fields, such as linear programming, simulation, numerical computation, etc. This also implies that data mining models have to be constructed and evaluated taking care of their application context and their relation with several other models, rules, constraints and goals.

As future work, many ideas follow from this work. For instance, we plan to develop new evaluation metrics which consider the quality of a predictive data mining model not only as the accuracy of its estimations given the inputs, but also its quality when the model has to be inverted. For instance, in our experiments we have found that regression trees are better than linear regression, and, in some cases, better than the direct classification decision

trees approach. This suggests the development of evaluation metrics which can be used to select the best models before application. Another issue for future analysis is the use of more efficient techniques to compute the curves and the envelopes when we have a high number of items and customers, since the combinations are quadratic.

In this work, we only have one negotiable feature at a time, but we are studying the extension to multiple negotiable features. When we have only one negotiable feature we have two dimensions (the negotiable feature and the probability). In the case of multiple negotiable features we have one dimension for each negotiable feature plus one (the probability). For example, if we have two negotiable features, we have three dimensions, and instead of having curves, in this case, we have surfaces. MEP and MGO strategies can be applied to multiple negotiable features without any problem, but the BLEP strategy needs changes, because each negotiable feature is monotonic, but all the negotiable features could not been monotonic at the same time, and this is a problem for the normalisation phase of the BLEP algorithm. Finally, other areas of possible research are the enrichment of the scenario with counteroffers from the customer, perishable and/or limited stocks, as well as the application to other areas outside CRM, such as medical research.

References

1. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.
2. A. Bella, C. Ferri, J. Hernández-Orallo, and M.J. Ramírez-Quintana. Joint cutoff probabilistic estimation using simulation: A mailing campaign application. In *IDEAL*, volume 4881 of *LNCS*, pages 609–619. Springer, 2007.
3. M.J.A. Berry and G.S. Linoff. *Mastering Data Mining: The Art and Science of Customer Relationship Management.* Wiley, 1999.
4. A. Berson, S. Smith, and K. Thearling. *Building Data Mining Applications for CRM.* McGraw Hill, 2000.
5. D.P. Bertsekas. Dynamic programming and optimal control, 3rd Edition. 2005.
6. M. Better, F. Glover, and M. Laguna. Advances in analytics: integrating dynamic data mining with simulation optimization. *IBM J. Res. Dev.*, 51(3):477–487, 2007.
7. N. Bradley. *Marketing Research. Tools and Techniques.* Oxford University Press, 2007.
8. J. Carbo and A. Ledezma. A machine learning based evaluation of a negotiation between agents involving fuzzy counter-offers. In *AWIC*, pages 268–277, 2003.
9. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition (Stochastic Modelling and Applied Probability).* Springer, 1997.
10. C. Ferri, P.A. Flach, and J. Hernández-Orallo. Improving the AUC of probabilistic estimation trees. In *14th European Conference on Machine Learning, Proceedings*, pages 121–132. Springer, 2003.
11. C. Ferri, J. Hernández-Orallo, and R. Modroiu. An experimental comparison of performance measures for classification. *Pattern Recogn. Lett.*, 30(1):27–38, 2009.
12. D. Fudenberg and J. Tirole. *Game theory.* MIT Press, 1991.
13. A. Gustafsson, A. Herrmann, and F. Huber. Conjoint analysis as an instrument of market research practice. *Conjoint measurement*, pages 3–30, 2000.
14. J. Han and M. Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, 2006.
15. J. J. Heckman. *Sample Selection Bias as a Specification Error.* Econometrica, 1979.

16. N.R. Jennings, P. Faratin, A.R. Lomuscio, S. Parsons, M.J. Wooldridge, and C. Sierra. Automated negotiation: Prospects, methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.
17. C. Kilian. *Modern Control Technology*. Thompson Delmar Learning, 2005.
18. S. Li, B. Sun, and R.T. Wilcox. Cross-selling sequentially ordered products: An application to consumer banking services. *Journal of Marketing Research*, 2005.
19. N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association (American Statistical Association)*, 1949.
20. B. Padmanabhan and A. Tuzhilin. On the use of optimization for data mining: Theoretical interactions and ecrm opportunities. *Management Science*, 49(10, Special Issue on E-Business and Management Science):1327–1343, 2003.
21. M. Peterson. *An Introduction to Decision Theory*. Cambridge University Press, 2009.
22. A. Prinzie and D. Van den Poe. Constrained optimization of data-mining problems to improve model performance: A direct-marketing application. *Expert Systems with Applications*, 29(3):630–640, 2005.
23. A. Prinzie and D. Van den Poel. Exploiting randomness for feature selection in multinomial logit: A crm cross-sell application. In *Advances in Data Mining*, volume 4065 of *Lecture Notes in Computer Science*, pages 310–323. Springer Berlin / Heidelberg, 2006.
24. M. L. Puterman. *Markov Decision Processes*. Wiley, 1994.
25. J. R. Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348. World Scientific, 1992.
26. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.
27. S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.
28. R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. The MIT press, 1998.
29. I. Vetsikas and N. Jennings. Bidding strategies for realistic multi-unit sealed-bid auctions. *Autonomous Agents and Multi-Agent Systems*, 21:265–291, 2010. 10.1007/s10458-009-9109-6.
30. E.W. Weisstein. *CRC concise encyclopedia of mathematics*. CRC Press, 2003.
31. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Elsevier, 2005.
32. M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons Ltd, 2002.
33. S. Zhang, S. Ye, F. Makedon, and J. Ford. A hybrid negotiation strategy mechanism in an automated negotiation system. In *ACM Conference on Electronic Commerce*, pages 256–257. ACM, 2004.