Data Mining Strategies for CRM Negotiation Prescription Problems [⋆]

A. Bella, C. Ferri, J. Hernández-Orallo, and M.J. Ramírez-Quintana

DSIC-ELP, Universidad Politécnica de Valencia, Camí de Vera s/n, 46022 Valencia, Spain

Abstract. In some data mining problems, there are some input features that can be freely modified at prediction time. Examples happen in retailing, prescription or control (prices, warranties, medicine doses, delivery times, temperatures, etc.). If a traditional model is learned, many possible values for the special attribute will have to be tried to attain the maximum profit. In this paper, we exploit the relationship between these modifiable (or negotiable) input features and the output to (1) change the problem presentation, possibly turning a classification problem into a regression problem, and (2) maximise profits and derive negotiation strategies. We illustrate our proposal with a paradigmatic Customer Relationship Management (CRM) problem: maximising the profit of a retailing operation where the price is the negotiable input feature. Different negotiation strategies have been experimentally tested to estimate optimal prices, showing that strategies based on negotiable features get higher profits.

1 Introduction

In data mining, problem features (or attributes) have been usually classified as input and output features. A problem is said to be supervised if it has output features, and it is said to be unsupervised if it does not have output features. Input features can be of many kinds: numerical, nominal, structured, etc. In fact, many data mining methods have been specialised to specific kinds of input features. In supervised learning, it is usually assumed that the goal of a model is to predict an output value given an input. Consequently, a function is learned from inputs to outputs, which is eventually applied to new cases.

However, in many application areas not all input feature values are given. This does not mean that they are unknown (i.e., null), but that they can be modified or fine-tuned at prediction time. Consider a typical data mining problem: a loan granting model where loans are granted or not according to a model which has been learnt from previous customer behaviours. It is generally assumed that given the input feature values (the customer's personal information, the loan amount, the operation's data, etc.) the model will provide an output (yes/no, a probability, a profit, etc.). But in real scenarios, the decision of whether the loan

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

must be granted or not might change if one or more of the input feature values can be changed. Perhaps, a loan cannot be granted for 300,000 euros, but it can be granted for 250,000 euros. If the customer asks the bank's clerk "what is the maximum amount you can grant for this operation?", we have a sort of *inverse* problem. Since the model is not generally an analytical one (it is not generally a linear regression model but a neural network, SVM, decision tree or other kind of difficult-to-inverte models), the only possibility to give a precise answer to the customer is to try all the possible input combinations for the loan amount and find out the maximum value which is granted. After this inefficient and ad-hoc process, the clerk can give an answer such as: "According to our loan models, we can grant a maximum of 278,304 euros". Apart from this inefficiency, there is another more serious problem: the clerk knows that the model will give a negative answer to any amount above this maximum, for instance, 278,305 euros, which makes this loan a quite risky one.

This typical example shows that some input features are crucial in the way that they can be freely modified at prediction time. The existence of these special attributes makes it quite inefficient to develop a classification/regression model in the classical way, since whenever there is a new instance hundreds of possible values have to be tried for the special attribute in order to see which combination can attain the maximum profit, or, as the previous example, the maximum risk. Additionally, these features are frequently associated to negotiation scenarios, where more than one attempt or offer have to be made, by suitably choosing different values for this special input feature.

In this paper we analyse these special features that we call "negotiable features", and how they affect data mining problems, its presentation and its use for confidence probability estimation and decision making. As we will see, we can exploit the relation between these input features and the output to change the problem presentation. In this case, a classification problem can be turned into a regression problem over an input feature [1]. Also, we present a first general systematic approach on how to deal with these features in supervised models and how to apply these models in real negotiation scenarios where there are several attempts for the output value.

The paper is organised as follows. In Section 2 we focus on classification problems with one numerical negotiable feature, since this is the most frequent and general case, and it also includes prototypical cases, when the negotiable feature is price or time. We present a general approach to the inversion problem which transforms the classification problem into a regression one, where the negotiable feature is placed as the output. In Section 3, we describe a specific real scenario (an estate agent's) where negotiation can take place. We also develop some negotiation strategies using the previous approaches in Section 4. In Section 5, we experimentally evaluate models and negotiation strategies with a baseline approach. Section 6 includes the conclusions and future work.

2 Inverting Problem Presentation

As we have mentioned in the introduction, there are many data mining problems where one or more input attributes, we call negotiable features, can be modified at application time. 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.

The inversion problem can be defined as follows. Consider a supervised problem, where input attribute domains are denoted by X_i, $i \in \{1, \ldots, m\}$, and the output attribute domain is denoted by Y. We denote the target (real) function as $f : X_1 \times X_2 \times \ldots \times X_m \to Y$. Values for input and output attributes will be denoted by lowercase letters. Hence, labelled instances are then tuples of the form $\langle x_1, x_2, \ldots, x_m, y\rangle$ where $x_i \in X_i$ and $y \in Y$. The inversion problem consists in defining the function $f^I : X_1 \times \ldots \times X_{i-1} \times Y \times X_{i+1} \times \ldots \times X_m \to X_i$, where X_i is the negotiable feature. 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 fixed the delivery time.

In the inverting problem the property that we call *sensitive* is satisfied. Fixing the values of all the other input attributes $X_j \neq X_i$ for at least n examples from the dataset D ($n \leq |D|$ being n determined by the user depending on the problem, the presence of noise, ...), there are two different values for X_i producing different output values.

We also assume a monotonic dependency between the input attribute X_i and the output. This dependency is defined under the assumption that there is a strict total order relation for the output, denoted by $\prec$, such that for every two different possible values $y_a, y_b \in Y$, we have that either $y_a \prec y_b$ or $y_b \prec y_a$. This order usually represents some kind of profit, utility or cost. For numerical outputs, $\prec$ is usually the order relation between real numbers (either $<$ or $>$, depending on whether it is a cost or profit). For nominal outputs, $\prec$ 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 $NEG \prec POS$. For more than two classes, the order relation can be derived from the cost of each class. Analogously, there is also a total order relation for the input denoted as $\preceq$. Based on this order, we can establish a monotonic dependency between the input and the output features. Thus, $\forall a, b \in X_i$, if $a \preceq b$ then $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) \prec f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m)$ (monotonically increasing) or $\forall a, b \in X_i$, if $a \preceq b$ then $f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_m) \succeq f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_m)$ (monotonically decreasing).

The inversion problem is well-known [1] 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.

But if we take a closer look at these issues, we see that although relevant, there is still a possibility behind this problem presentation change. First, many regression techniques work well for inconsistent examples, so this is not 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. We extend this idea further below.

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 $p(POS|\langle x_1, \dots, x_{i-1}, a, x_{i+1}, \dots, x_m\rangle)$, for any possible value $a \in X_i$. From the inverted (regression) problem, we get: $\hat{a} = f^I(x_1, \dots, x_{i-1}, POS, x_{i+1}, \dots, x_m)$. 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, that is $p(POS|\langle x_1, \dots, x_{i-1}, \hat{a}x_{i+1}, \dots, x_m\rangle) = 0.5$.

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 standard error (mean absolute error, mae, on the training set) as standard deviation σ. In other words, we use $N(\hat{a}, mae^2)$. 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.

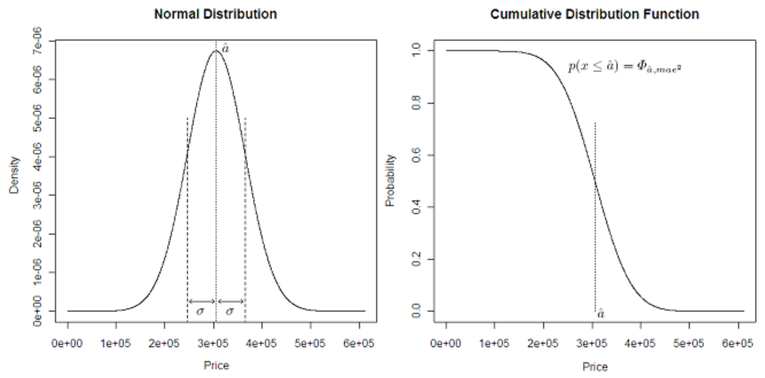

Fig. 1. Left: Example of a normal distribution $\hat{a} = 305,677.9$ and $\sigma = 59,209.06$. Right: Associated cumulative distribution function.

From here, we can derive the probability for any possible value a as the cumulative distribution function derived from the above normal, i.e., $\Phi_{\hat{a},mae^2}$.

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 standard error as standard deviation. We call this model *negotiable feature model*.

3 Negotiation using Negotiable Feature Models: A Real Scenario

We are going to illustrate the approach we have presented in the previous section in a real example. In particular, we have studied the problem of retailing, where the (negotiable) input feature is the price (denoted by π) and the problem is a classification problem (buying or not).

We present an example using real data from an estate agent's, which sells flats and houses they have in their portfolio. We have several conventional attributes describing the property (squared metres, location, number of rooms, etc.), and a special attribute which is our negotiable feature, price. We will use the term "product" for properties to make it clear that the case is directly extensible to virtually any retailing problem where price is negotiable.

We start with the simplest negotiation scenario, where there are only one seller and one buyer who both negotiate for one product. One buyer is interested in 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.In fact, 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. Moreover, the relation between price and the class order $NEG \prec POS$ is monotonically decreasing.

Additionally, in our problem, the seller has a "minimum price" (denoted by π_{min}), which corresponds to the price that the owner has set when the product was included in the portfolio plus a quantity that the seller sets as fixed and variable costs. 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. Finally, the profit obtained by the product will be the difference between the selling price minus the minimum price: $Profit(\pi) = \pi - \pi_{min}$.

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

$$f(x_1, \ldots, x_{i-1}, \pi_{max}, x_{i+1}, \ldots, x_m) = POS$$

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

In other words, the use for the model is: "Which is the maximum price that I can sell this product to this customer?". Logically, the higher the price the lower the probability, so the goal is more precisely 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 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 (1) as:

$$E\_Profit(\pi) = \hat{p}(POS|\pi) \cdot Profit(\pi) \quad (2)$$

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

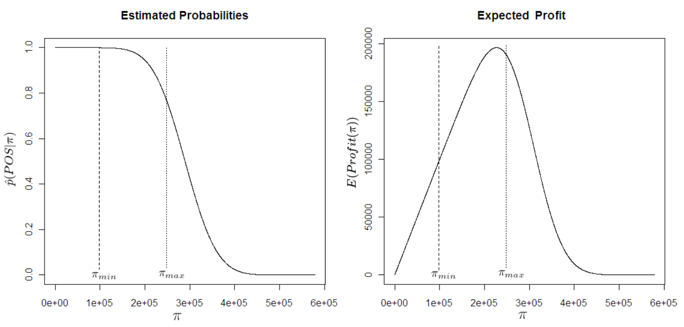

Fig. 2. Left: Example of estimated probabilities. Right: Associated expected profit. The minimum and maximum price are also shown.

So, if we have a model which estimates probabilities for the positive class, we can use formula (2) 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. In Figure 2 an example of the plots that are obtained for the estimated probabilities and expected profit is shown.

But this is the case where we have one bid (one offer). In many negotiation scenarios, we have the possibility of making several bids, as in bargaining. In this situation, it is not so direct how to use the model in order to set a sequence of bids to get the maximum overall expected profit. For instance, if we are allowed three bids, the overall expected profit of a sequence of bids is defined as:

$$E\_Profit(\langle \pi_1, \pi_2, \pi_3 \rangle) = \hat{p}(POS|\pi_1) \cdot Profit(\pi_1) + (1 - \hat{p}(POS|\pi_1)) \cdot \hat{p}(POS|\pi_2) \cdot Profit(\pi_2) + (1 - \hat{p}(POS|\pi_1)) \cdot (1 - \hat{p}(POS|\pi_2)) \cdot \hat{p}(POS|\pi_3) \cdot Profit(\pi_3),$$

where $\pi_1 > \pi_2 > \pi_3 \geq \pi_{min}$.

4 Negotiation Strategies

In the scenario described in Section 3 the seller is the agent who uses the negotiable feature models to guide the negotiation, while the buyer can only make the decision of buying or not the product.

When we have one possible offer, it is not sensible to set the price at the maximum price that our model predicts it can be sold, because in many cases, because of the prediction error, it will be overestimated and we will not sell the product. On the contrary, selling at the minimum price ensures that we sell as many products as possible, but we get minimum profit as well. In Section 3 we saw that an appropriate way of doing this is by using the expected profit.

Obviously, if the maximum price for the buyer is lower than the minimum price for the seller, the product is not sold. We will exclude these cases, since any strategy is not going to work well for them and it is not going to make any difference to include them or not in terms of comparison.

When we have more than one possible offer, we start with a first offer and if the price is less or equal than a price which is accepted by the buyer, s/he will buy the product. Otherwise, the seller can still make another offer and follow the negotiation.

It is clear that there exists an optimum solution to this problem when the seller can make "infinite" offers to the buyer, but it is inefficient and unfeasible. This idea consists in beginning the negotiation with a very high offer and make offers of one euro less each time, until the (patient and not very intelligent) buyer purchases the product or until the price of the product is the minimum price. In this case the product would be sold by its maximum price, because the buyer purchases the product when the price offered was equal to the maximum price considered by the buyer.

In what follows we propose several strategies. One is the "baseline" method which is typically used in real estate agent's. For cases with one single bid, we introduce the strategy called "Maximum Expected Profit" (MEP), which is just the application of the expected profit as presented in the previous section. 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:

- Baseline method (1 bid or N bids). One of the simplest methods to price a product is to increase 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 if we obtain that the best α is 0.4, it is expected that the best profit will be obtained increasing in 40% the minimum price of the properties. 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, for 3 bids and the previous sample the three values of α for three bids would be 50%, 40% and 30%. Therefore, the first offer would be an increase of 50% over the minimum price of the product, the second an increase of 40% and the third an increase of 30%.
- 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. $\pi_{MEP} = argmax_{\pi}(E\_Profit(\pi))$.
- 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. In the case

of the probability of buying which is associated to the price is the maximum probability, it 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 price. The pseudo-code is in Algorithm 1.

– Maximum Global Optimisation (MGO) strategy (N bids). The objective of this strategy is to obtain the N offers that maximise the expected profit by generalising the formula that we have presented in Section 3:
$\pi_{MGO} = argmax_{\langle \pi_1,\ldots,\pi_N \rangle}(E\_Profit(\langle \pi_1, \ldots, \pi_N \rangle)) = argmax_{\langle \pi_1,\ldots,\pi_N \rangle}(\hat{p}(POS|\pi_1) \cdot Profit(\pi_1) + (1 - \hat{p}(POS|\pi_1)) \cdot \hat{p}(POS|\pi_2) \cdot Profit(\pi_2) + \ldots + (1 - \hat{p}(POS|\pi_1)) \cdot \ldots \cdot (1 - \hat{p}(POS|\pi_{N-1})) \cdot \hat{p}(POS|\pi_N) \cdot Profit(\pi_N))$.
Getting 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 with a sufficient number of tuples to get the values for the prices that maximise the expected profit.

```
Algorithm 1: BLEP strategy
Require: $N$, $epf$ (estimated probability function or curve)
Ensure: $\pi_{BLEB}$
  $\forall x, epf(x) \leftarrow \hat{p}(POS|x)$
  $\pi_1 \leftarrow \pi_{MEB}$
  $\pi \leftarrow \pi_1$
  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
      $\pi_i \leftarrow \pi_{MEB}$
      $\pi \leftarrow \pi_i$
    else
      $\pi_i \leftarrow \pi \div 2$
      $\pi \leftarrow \pi_i$
    end if
  end for
  $\pi_{BLEB} \leftarrow \langle \pi_1, \ldots, \pi_N \rangle$
```

5 Experiments

5.1 Experimental Settings

Experiments have been performed by using real data collected from an estate agent's. We have information of 2,800 properties (flats and houses) that were sold in the last months, for which we have the following attributes ("district", "number of rooms", "square metres" and the "owner's price"). The "owner's price" is the price which the owner wants to obtain for the property.

In the experiments we have assumed that the "owner's price" is some kind of "market price" and we have considered that it is the "maximum price". Although it is not always true because in some cases the buyer could have paid more than this for the property.

We have randomly split the dataset 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 2 to the data. In particular we have used a $J48$ decision tree[1] (with Laplace correction and without pruning) implemented in the data mining suite WEKA [3]. Since the predicted probability curve given by a classifier (such as the J48 classifier) typically shows discontinuities and strong steps when varying a negotiable feature, we have smoothed it with a low-pass filter with Bartlett overlapping window [2]. The parameter of the window has been set to the "minimum price" divided by 400. The "inversion problem" solution has been implemented with the *LinearRegression* and $M5P$ regression techniques, also from WEKA.

These three learning techniques have been used to guide the three negotiation strategies explained in Section 4 (for the MGO strategy we used a Montecarlo approach using 1,000 random triplets) and they are compared to the two baseline methods also mentioned in Section 4. In the experiments the number of bids is either one or set to three, i.e., $N = 3$. Summing up, we have nine negotiation methods based on learning techniques and also two baseline methods (without learning process) using the best possible α (80% for one bid and the triplet $\langle 100\%, 80\%, 60\% \rangle$ for three bids).

5.2 Experimental Results

In Table 1 we can observe the results obtained for each method, in terms of number of sold properties, total sold price (in euros) and total profit (in euros).

Table 1. Results obtained by the negotiation strategies, baseline methods and reference methods (minimum and maximum profit). Sold price and profit measured in euros.

Method	**Sold flats**	**Sold price**	**Profit**
All flats sold at π_{min}	2,520	356,959,593	0
All flats sold at π_{max}	2,520	712,580,216	355,620,623
1 bid			
Baseline (80%)	1,411	200,662,464	89,183,317
MEP ($J48$)	1,360	302,676,700	129,628,471
MEP (*LinearRegression*)	1,777	354,973,300	159,580,109
MEP ($M5P$)	1,783	358,504,700	161,736,313
3 bids			
Baseline (100%, 80%, 60%)	1,588	264,698,467	124,483,288
BLEP ($J48$)	1,940	382,921,400	173,381,116
BLEP (*LinearRegression*)	2,056	400,953,200	174,832,025
BLEP ($M5P$)	2,063	404,009,700	176,874,221
MGO ($J48$)	1,733	390,529,770	176,020,611
MGO (*LinearRegression*)	1,918	475,461,200	232,600,223
MGO ($M5P$)	1,906	476,171,900	234,259,509

As we see in Table 1 all the negotiation methods outperform the baseline methods. For one bid, MEP is clearly better than the baseline method. For three bids, both BLEP and MGO are much better than the baseline method. Overall, MGO makes a global optimisation and hence get better results.

[1] In this problem, we only have data of sold properties (positive class), therefore, we have generated examples of the negative class with a price higher than the "owners's price".

On the other hand, the regression techniques outperform the $J48$ decision tree, so, for this problem the solution of inverting problem presentation outperforms the improved classifier solution. This is especially dramatic for MGO, which is the method that depends most on a good probability estimation. This means that the inversion problem using a normal distribution to get the estimated probabilities turns out to be a very useful approach.

6 Conclusions

This paper introduces a number of new contributions in the area of data mining and machine learning which can be useful for many application areas: retailing, control, prescription, and others where negotiation or fine-tuning can take place. Nonetheless, the implications can affect other decision-making problems where data mining models are used.

The first major contribution is the analysis of the relation between negotiable features and problem presentation, and more specifically the inversion problem which happens naturally when we have to modify or play with the negotiable attribute. We have seen that using the monotonic dependency we can change of problem presentation to do the inversion problem (such as changing a classification problem into a regression), where the original problem is now indirectly solved by estimating probabilities using a normal distribution.

The second major contribution is its application to real negotiation problems. We have developed several negotiation strategies and we have seen how they behave for one or more bids in a specific problem of property selling. We have shown that our approach highly improves the results of the classical baseline method (not using data mining) which is typical in this area. In the end, we show that the change of problem presentation (from classification into regression problem, using the negotiable feature, price, as output) gets the best results for the case with one bid but also for the case with three bids.

Since this work introduces new concepts and new ways of looking at some existing problems, many new questions and ideas appear. For instance, we would like to analyse how to tackle the problem when there are more than one negotiable feature at a time, especially for the inversion problem approach (since we would require several models, one for each negotiable feature).

Finally, more complex negotiation strategies and situations can be explored in the future: the customer can counter-offer, several customers, etc.

References

1. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition (Stochastic Modelling and Applied Probability).* Springer, 1997.
2. E.W. Weisstein. *CRC concise encyclopedia of mathematics.* CRC Press, 2003.
3. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Elsevier, 2005.