Preference Learning from Qualitative Partial Derivatives

Jure Žabkar, Martin Možina, Tadej Janež, Ivan Bratko, and Janez Demšar

University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, SI-1000 Ljubljana, Slovenia
jure.zabkar@fri.uni-lj.si

Abstract. We address the problem of learning preference models from data containing implicit preference information. The information about preferences is expressed as a result of a decision or action in a given state in context of other attributes. We propose an algorithm that observes the change of probability of a target class (desired result) w.r.t. the change in the values of the attribute describing the decision or action. We generalize the notion of a partial derivative by defining the *probabilistic discrete qualitative partial derivative (PDQ PD)*. PDQ PD is a qualitative relation between the target class c and a discrete attribute, given as a sequence of attribute values a_i in the order of $P(c|a_i)$ in a local neighbourhood of the reference point. In a two stage learning process we first compute PDQ PD for all examples in the training data, and then generalize over the entire data set using an appropriate machine learning algorithm (e.g. a decision tree). The induced preference model explains the influence of the attribute values on the target class in different subspaces of the attribute space.

1 Introduction

The typical task of preference learning is to induce models for prediction of preferences, based on learning data which includes descriptions of examples and their preferences. Consider, for example, a flight carrier desiring to predict the food preferences of their passengers. John, a white middle-aged male from Wales, might prefer non-vegetarian food over vegetarian over snacks over drink-only service. Preference learning generalizes such cases into models.

Actual data often does not look like this. Passengers, for instance, are not asked which food they prefer but rather whether they were satisfied with what they were given or not. We may know that John, a white male from Wales, age 56, who was given veal and rice topped with a delicate mustard sauce, called it a good meal, while Jack, a white middle-aged male from Scotland did not appreciate the pretzels and sparkling water he was provided on his flight. In this paper we present an algorithm for generalizing the preference relations from this kind of data. We assume the learning examples described by discrete attributes (gender, age, type of food) and classified into ordered classes (*disliked*, *OK'd*, *liked* the food). The algorithm can estimate the preferences for each example with regard to each attribute. For instance, it can tell us that John, the white middle-aged man from Wales, prefers non-vegetarian over sweets over vegetarian over drink-only:

$$\text{drink-only} \prec \text{vegetarian} \prec \text{sweets} \prec \text{non-vegetarian}.$$

Other attributes can also be used to derive the preferences. For instance, we may learn that white middle-aged inhabitants of Wales who are given vegetarian lunch are more likely to be satisfied with it if they are female. This is a sex-preference relation for a certain combination of attributes. Or, for instance, it can tell us that the satisfaction of white middle-aged men with pretzels does not depend on whether they are Scottish or Welsh, which can be seen as ethnic preference (or indifference) for a certain kind of food.

Our venture point in development of the presented algorithm is qualitative modelling. A branch of qualitative modelling considers learning qualitative models from numerical data [4,25,26]. It is a technique similar to regression modelling, except that instead of numerical predictions it predicts the direction of change (increase, decrease, no change) of the target variable. Such models are preferred over regression models when exact numerical models are difficult to obtain or difficult to use due to unreliable measurements, and, in particular, when the task is to predict the effect that a change in input variables will have on the observed variable. For instance, a qualitative model may describe the conditions under which a decrease of interest rates will stimulate investments and how will this affect the unemployment rate. While exact numerical models for this problem are elusive, qualitative relations between these variables may be easier to describe.

In this paper we extend this type of reasoning to categorical domains. We describe an algorithm Qube [24] for calculating qualitative preferences, a type of qualitative models describing the influence of a categorical (e.g. nominal) variable on a target class probability. Instead of dealing with direction of change, we rank attribute values so that the probability of the target class increases. By ranking, we drop all numerical information (probabilities). Finally, as we generalize over the entire data set, the induced qualitative model describes the conditions under which certain attribute values have greater/lower influence on the target class probability.

2 Methods

We will first provide a definition of probabilistic discrete qualitative partial derivatives based on conditional probabilities of the target class given the attribute values. The computation of these probabilities from data requires selecting proper subsets of examples, which we will describe next. Finally, we show how to combine the computed probabilities into partial derivatives and use them to induce qualitative models.

2.1 Probabilistic discrete qualitative partial derivative

Derivative of a function $f(x)$ at a certain point x_0, $f'(x_0)$, tells us, informally, the change of the function value corresponding to a certain (small) change of the value of the function's argument, *i.e.*

$$f(x_0 + \Delta x) - f(x_0) \approx f'(x_0)\Delta x. \tag{1}$$

For functions of multiple arguments, *e.g.* $f(x_1, x_2, \ldots, x_n)$, we compute the derivative w.r.t. each argument x_i separately and denote it by $\partial f / \partial x_i$.

Qualitative derivatives are similar to ordinary derivatives except that they give only the direction of change, that is, whether the function will increase or decrease when its argument increases. Qualitative derivative of a function is positive (negative, zero) if the continuous derivative is positive (negative, zero),

$$\frac{\partial_Q f}{\partial_Q x} = \text{sgn}\frac{\partial f}{\partial x}. \tag{2}$$

Now consider a multivariate distribution which assigns a probability y to each element of Cartesian product of $\mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \times \mathcal{A}_n$. In machine learning, $(a_1, a_2, \ldots, a_n) \in \mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \times \mathcal{A}_n$ can be values of discrete attributes $A_1, A_2, \ldots, A_n$ describing an example (we will call this example a *reference example*), and y is the probability that such an example belongs to some target class c,

$$y = p(c|a_1, a_2, \ldots, a_n). \tag{3}$$

Recall that qualitative derivative of f w.r.t. x_i computed at $(x_1, x_2, \ldots, x_n)$ tells whether a certain change of x_i will *increase or decrease the function value* if other arguments remain constant. Let the *probabilistic discrete qualitative partial derivative* at $(a_1, \ldots, a_n)$ with respect to A_i tell whether a change of value of the attribute A_i from a_i to a_i' will *increase or decrease the probability* of the target class:

$$\frac{\partial_Q f}{\partial_Q A_i : a_i \to a_i'}(a_1, \ldots, a_n) = \\ \begin{cases} +, p(c|a_1, \ldots, a_i, \ldots, a_n) < p(c|a_1, \ldots, a_i', \ldots, a_n) \\ \circ, p(c|a_1, \ldots, a_i, \ldots, a_n) = p(c|a_1, \ldots, a_i', \ldots, a_n) \\ -, p(c|a_1, \ldots, a_i, \ldots, a_n) > p(c|a_1, \ldots, a_i', \ldots, a_n). \end{cases} \tag{4}$$

Let us define a partial order on set $\mathcal{A}_i$, with respect to fixed values of a_j for all $j \neq i$

$$a_i \prec a_i' \Leftrightarrow \\ \Leftrightarrow p(c|a_1, \ldots, a_i, \ldots, a_n) \leq p(c|a_1, \ldots, a_i', \ldots, a_n) \tag{5}$$

This allows us to rewrite (4) as

$$\frac{\partial_Q f}{\partial_Q A_i : a_i \to a_i'}(a_1, \ldots, a_n) = \begin{cases} +, & a_i \prec a_i' \\ \circ, & a_i = a_i' \\ -, & a_i \succ a_i'. \end{cases} \tag{6}$$

The derivative $\partial_Q f / \partial_Q A_i : a_i \to a_i'$ for any pair a_i and a_i' can thus be described by a partial ordering of attribute values $\mathcal{A}_i$.

2.2 Computation of conditional probabilities

To compute the derivative we have to estimate the conditional probability $p(c|a_1, \ldots, a_n)$ at a single point $(a_1, a_2, \ldots, a_n)$ from data sample. This probability cannot be estimated directly, for instance, by relative frequencies, since there may be only a few or

even no examples in the data which match the condition part. We also cannot use a naive Bayesian method since it would reduce the PDQ PD to comparison of $p(c|a_i)$ and $p(c|a_i')$, cancelling out all the terms corresponding to the values of other attributes. This is easily explained: the naive Bayesian assumption of conditional independence of attributes given the class implies that the derivative $\partial_Q f / \partial_Q A_i : a_i \rightarrow a_i'$ is constant on the entire attribute space.

The problem requires a semi-naive Bayesian approach. We will replace the condition in $p(c|a_1, \ldots, a_n)$ with a relaxed condition $\mathcal{D} \subseteq \{a_1, a_2, \ldots, a_n\}$, where $\mathcal{D}$ will include only the attribute values which are conditionally dependent on a_i given the class. Ignoring the other, conditionally independent values do not change the computed derivative (see the proof in Appendix).

To construct the set of conditions $\mathcal{D}$ we will use a greedy approach: we start with an empty set $\mathcal{D}$ and iteratively add the most dependent value. Let e_j represent an event that attribute A_j on a certain example has a value a_j. Let event v represent values of attribute A_i on an example ($v \in \mathcal{A}_i$). We need to test the hypothesis that e_j and v are conditionally independent given the class and a set of existing conditions $\mathcal{D}$ (that is, knowing the class of an example and knowing that the example satisfies $\mathcal{D}$, the probability distribution for values of A_i is independent of whether the value of j-th attribute equals a_j or not). In each step of the algorithm we test the dependence between v and e_j, find e_j which most strongly violates the independence and add the corresponding a_j to $\mathcal{D}$.

The independence is tested using the standard χ^2 test. Separate tables with $2 \times |\mathcal{A}_i|$ cells are constructed for class c and its complement. We compute the expected absolute frequencies for the first table as $n(c, \mathcal{D})p(a_j|c, \mathcal{D})p(v|c, \mathcal{D})$ and $n(c, \mathcal{D})(1 - p(a_j|c, \mathcal{D}))p(v|c, \mathcal{D})$, where $v \in \mathcal{A}_i$ and $n(c, \mathcal{D})$ is the number of examples in class c which satisfy the conditions $\mathcal{D}$. Frequencies for the complement of c are computed analogously. The sum of χ^2 statistics for both tables is distributed according to χ^2 distribution with $2(|\mathcal{A}_i| - 1)$ degrees of freedom.

For each a_j we compute its corresponding p-value and select the one with the lowest value. We stop the selection procedure when the lowest p-value is above the specified threshold or when the number of examples matching the conditions $\mathcal{D}$ falls below the given minimum. This is needed to ensure the reliability of χ^2 statistics and of estimated conditional probabilities. Our use of p-value does not require adjustments for multiple hypotheses testing since the p-value is used only as a stopping criteria and not to claim the significance of the alternative hypothesis.

After choosing a set of conditions $\mathcal{D}$, we compute $p(c|v, \mathcal{D})$ for all $v \in \mathcal{A}_i$ using relative frequency, Laplace estimate or m-estimate [6] on examples matching $\mathcal{D}$.

Note that the χ^2 test is performed over all values of A_i and not only a_i and a_i'. This lets us use the same set of conditions $\mathcal{D}$ for all derivatives with respect to A_i at a certain reference example and ensures that probabilities $p(c|a_1, \ldots, a_i, \ldots, a_n)$ for all $a_i \in \mathcal{A}_i$ are comparable and thus useful for defining a partial ordering of a_i.

For another interpretation of this procedure, consider the definition of partial derivative of a continuous function:

$$\frac{\partial f}{\partial x_1}(x_1, \ldots, x_n) = \lim_{h \rightarrow 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(x_1, x_2, \ldots, x_n)}{h}. \qquad (7)$$

When computing the derivative w.r.t. x_1, we subtract the function value in two points where all arguments except x_1 are the same. If the function value at point $(x_1, \ldots, x_n)$ does not depend on, say, x_2, we could use different value of x_2 in the two terms. Omitting the values from (4) is similar to that: the χ^2 test is used to determine whether ignoring the difference between a_j and other values of the attribute A_j will affect the computation or not.

The proposed greedy procedure may seem naive, yet it works well in practice. We must also keep in mind that the selection of attributes needs to be fast since we have to run it for each point in which we compute the derivative, which rules out any advanced search for sets of dependent values.

2.3 Computation of derivatives

The partial order of $\mathcal{A}_i$ is determined by the order of the corresponding probabilities as defined in (4). To handle noisy data we will however treat two probabilities (and the corresponding values of A_i) as equal if they differ for less than a user-provided threshold.

For this we use hierarchical clustering of values with average linkage [23] using the difference of probabilities as distances. The clustering is stopped when the distance between the closest clusters is greater than 0.2.

For example, let A_i be a five-valued attribute with values v_1 to v_5. Probabilities $p(c|v_i, \mathcal{D})$ for these values equal 0.1, 0.2, 0.3, 0.5 and 0.6, respectively. Let the merging threshold be 0.2. We recognize v_1 and v_2 as equivalent and assign them the average probability of $(0.1 + 0.2)/2 = 0.15$. Next we merge v_4 and v_5, the average probability is $0.5 + 0.6 = 0.55$. Finally, we merge v_1 and v_2 with v_3; the average probability is $(0.1 + 0.2 + 0.3)/3 = 0.2$. We then stop since the difference between $p = 0.2$ (v_1 to v_3) and $p = 0.55$ (v_4 to v_5) exceeds the threshold of 0.2. The resulting partial ordering of A_i is $v_1 = v_2 = v_3 \prec v_4 = v_5$.

2.4 Induction of preference models

To induce a preference model with respect to a certain attribute A_i, we first compute the PDQ PD for the entire learning set: for each example, we compute the set of dependent values $\mathcal{D}$ and find the total ordering of attribute values $\mathcal{A}_i$ as explained in the previous two sections. We replace the original class labels with partial derivatives (that is, the total ordering) and induce a model for predicting the ordering. In principle, any learning algorithm can be used for this task.

3 Experiments

We will first show a toy experiment on the Titanic data set from UCI [13] repository. The second, more complex experiment will be conducted on a data set obtained from a billiards simulator.

Conditional probabilities for both are estimated using the m-estimate [6] with $m = 2$. We use a threshold of 0.2 for merging attribute values. Our reimplementation of the

C4.5 algorithm [9] is used to obtain qualitative models describing the relation between the target class and each attribute separately.

3.1 Titanic

Titanic data set consists of 2201 examples described by three attributes, *status (first, second, third, crew)*, *age (child, adult)* and *sex (male, female)*, and a class *survived (yes, no)*. We selected target class *survived = yes* and for each passenger computed the preference (with respect to survival) regarding the status. Finally, we used C4.5 to learn the preference model over the entire data set. Each passenger's status preferences express the ordering of the attribute values in which the probability of surviving is increasing.

For adult males, the probability of survival does not depend upon their status, therefore there is no status preference with respect to survival. For others, the probability is the smallest in the third, and thus "unpreferred" class. For women and boys, there is no difference between the other classes, while girls were more likely to survive in the second than in the first class, and should thus prefer the former. Based on the model, if the goal is to survive the sinking, the passenger's preference should be not to travel in the third class, and girls should, specifically, prefer the second class.

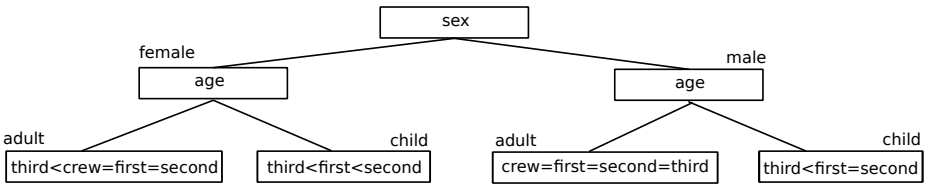


Fig. 1: Preference model for surviving induced from Titanic data set.

3.2 Billiards

Billiards is a common name for table games played with a stick and a set of balls, such as snooker or pool and their variants. The goals of the games vary, but the main idea is common to all: the player uses the stick to stroke the cue ball aiming at another ball to achieve the desired effect. The friction between the table and the balls, the spin of the cue ball and the collision of the balls combine into a very complex physical system [1]. However, despite of its complexity, an amateur player can still learn the basic principles of how to stroke the cue ball without knowing much about the physics behind it. In this case study we will use our method to induce a simple model, which could be used by a human player, for ranking shot types in different billiards positions.

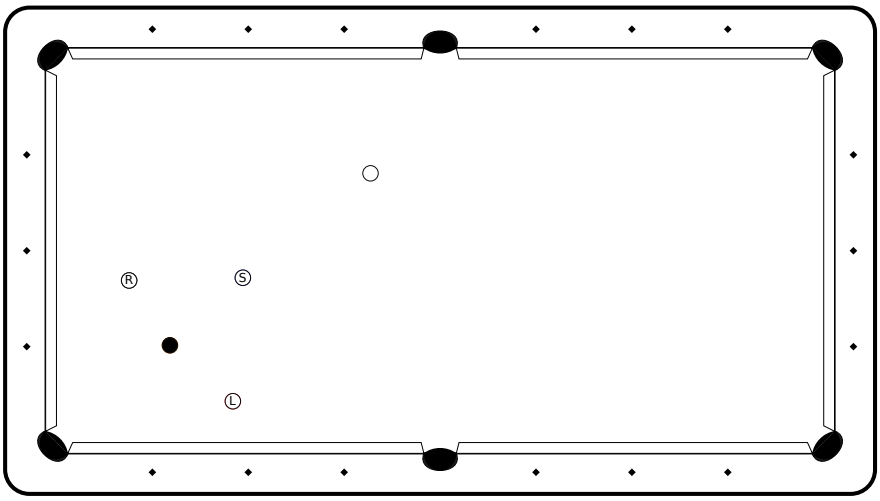


Fig. 2: Blocking the strokes. Ball S blocks a direct shot while balls L and R block left and right shots respectively.

Our goal is to learn shot preferences in different circumstances from simulated data. Although it is possible to pocket the black ball with several different type of shots, some shots are more appropriate than others thus increasing the probability of a successful shot. For example, if a hole, a black ball and a cue ball are collinear, a direct shot is preferred over hitting a rail first because it is much more difficult to correctly estimate the reflection angles so that the black ball would pocket. However, in the presence of other balls which may present an obstacle for a direct shot, a rail-first shot may be preferred.

We consider billiards positions with a cue ball (white colour), a black ball and three optional balls (L, S, and R) that may present an obstacle. The balls positions are fixed as shown in Figure 2.

There are several different types of shots in the game of billiards [1]. For the sake of simplicity, we will only consider direct shots and rail-first shots. In the former, the cue ball directly hits the black ball, without hitting anything else in between. In the latter, the cue ball first hits the rail (also called cushion) and only then the black ball. In our study, we defined 4 possible actions a player can make: *strong direct shot, weak direct shot, left rail-first shot* and *right rail-first shot*. Strong and weak direct shots (Figure 4a) differ in the force applied by the player, which affects the initial velocity of the cue ball. The difference between left and right rail-first shots is shown in Figures 4b and 4c. In the left-rail-first shot, the white ball will pass the black ball on the left, hit the rail, and then hit the black ball. Similarly, in the right-rail-first shot, the white ball will first pass the black ball on the right, hit the rail, and then hit the black ball.

We created a data set of 1000 shots (i.e. learning examples), where each example is described by the following four attributes:

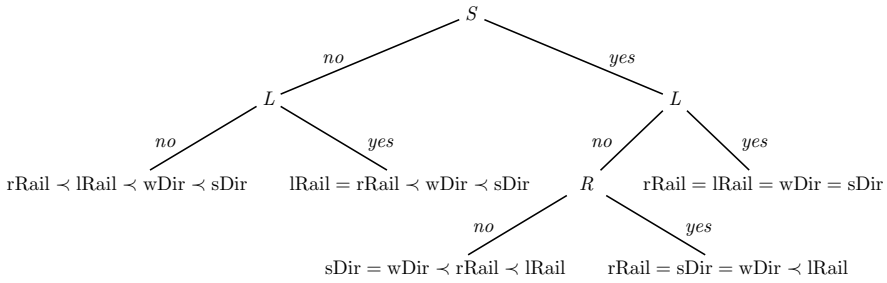


Fig. 3: The induced preference model suggesting the player the best action regarding the situation of the balls on the table.

- *S*: whether the ball S blocking a straight shot was present (values:yes/no),
- *L*: whether the ball L blocking a left shot was present(values:yes/no),
- *R*: whether the ball R blocking a right shot was present(values:yes/no),
- *ShotType*: the shot type (values:sDir,wDir,lRail,rRail),

and the class variable *black in pocket* with values *yes* and *no*. The goal (target class) in our method is thus '*black in pocket* = *yes*', namely, whether the shot forced the black ball into a pocket. The attribute values of each single example were randomly selected, while the value of the class was determined using the billiards simulator [22]. Each shot in the simulator is defined by: shot direction, stick elevation, shot velocity and shot follow. The most important property of a shot is its direction, as it has the dominant effect on the direction of the black ball after it is hit by the white ball. First, we computed the optimal stroke direction $opAng$ that results in pocketing the black ball. However, to account for human imprecision, we added uniform noise in the interval $[-0.2\,\mathrm{deg}, 0.2\,\mathrm{deg}]$. Therefore, the value of stroke direction in the simulator is set to a random value selected from the interval $[opAng - 0.2, opAng + 0.2]$. We would expect that more difficult shots are more prone to changes in the optimal setting. Similarly, other properties of the shot were randomly chosen from specified intervals. The *stick elevation* was from [0, 5] and the *shot follow* from [-0.1, 0.1]. The *shot velocity* was chosen from [.1, 2] for weak shots and from [3, 4] for strong shots.

We again used the C4.5 algorithm to obtain a qualitative model. The induced preference model is shown in Fig. 3. The interpretations of leaves of the tree are as follows:

- $S = \text{no} \wedge L = \text{no}$: Balls S and L are not on the table. In this case, strong direct shot is preferred over weak direct shot since the velocity of the black ball may not be sufficient for the black ball to reach the pocket using the weak shot. A weak shot is preferred over left rail-first shot which is preferred over right rail-first shot. In general, rail-first shots are less successful than direct shots due to distortions caused by cue ball hitting the rail. However, left rail-first shot is usually better since the path of the ball is shorter than the path in the right rail-first shot. It is irrelevant whether ball R is present or not because it can only block the right rail-first shot which has the lowest preference in any case.

- $S =$ no $\wedge$ $L =$ yes: Ball L is blocking the left rail-first shot which makes both direct shots preferred over the rail-first shots. The preference of direct shots is the same as above while both rail-first shots are equally preferred.
- $S =$ yes $\wedge$ $L =$ no $\wedge$ $R =$ no: Ball S is blocking direct shots which makes both rail-first shots preferable and both direct shots equally bad. Left rail-first shot is preferred over the right one for the same reason as above.
- $S =$ yes $\wedge$ $L =$ no $\wedge$ $R =$ yes: Balls S and R are blocking the shots. The left rail-first shot is preferred over all other shots since it is the only open shot.
- $S =$ yes $\wedge$ $L =$ yes: Balls S and L are blocking both direct and left rail-first shots. There are no shot preferences since all shots are equally difficult.

The induced model conforms with the expert opinion and common sense.

4 Related work

Preference learning [10,16,3] considers the problem of learning a preference model given learning examples as well as the preferences [2,7,5]. Our approach starts earlier and calculates the preferences for each learning example from data. While one could continue by using standard preference learning approaches [7,5], we use simple machine learning algorithms and treat preferences as values of a new class variable. Theoretically, it is possible that the number of class values exceeds a reasonable amount but it can be practically very well controlled by setting the threshold parameter (for joining the values) and the size of the neighbourhood of the reference example (a kind of smoothing the data).

The motivation for our work comes from the field of qualitative reasoning where Qube can be used for learning qualitative models from categorical data. Qualitative reasoning has been mostly concerned with qualitative physics [8,19,20,12,11]. In these works the model was provided by an expert and then used in qualitative simulations. There are only a few algorithms for automated induction of such models [17] and even these are limited to learning from numerical data [4,25]. There are, to the best of our knowledge, no algorithms for learning qualitative models from categorical data.

An important part of our method deals with relaxing the strong independence assumption of naive Bayesian approach. There exist a number of methods for this purpose, yet none fits our context. Kononenko introduced *semi-naive Bayes* [18] and Langley and Sage [21] proposed *Selective Bayesian Classifier*, a variant of the naive method that uses only a subset of the attributes in making predictions. Since their algorithm is only searching for subset of attributes that yields highest classification accuracy, it can not reveal attribute dependencies. Friedman and Goldszmidt introduced *tree augmented naive Bayes (TAN)* [15] which allows for attributes having another attribute as a parent in the bayesian network representation.

A lazy algorithm, named Locally Weighted Naive Bayes (LWNB) is proposed in [14]. LWNB relaxes the independence assumption by learning local models at prediction time. The models are learned on weighted set of training instances in the neighbourhood of the test instance. In LWNB, the test example neighbourhood is chosen using the k-nearest neighbours algorithm. A step further is the Lazy Bayesian Rules (LBR) algorithm [27]. LBR search of the local neighbourhood is not based on a global metric.

Instead, for each test example, LBR uses a greedy search to generate a Bayesian rule with an antecedent that matches the test example. The basic difference between these approaches and ours is that these methods are concerned with optimizing the accuracy of predictions and not with estimations of the chosen attribute's influence on the target class probability.

5 Conclusion

We have presented a novel approach to learning preference models. Its specific is that it learns the preference of values of an attribute, which leads to the preferred class in the context of other attributes. That is, in a given situation (as described by values of attributes) it tells which value of a particular attribute should increase the probability of the preferred class, where the preference of classes is known, *e.g.* to survive or to reach the goal of the game.

We demonstrated the method on the standard toy data set about Titanic and a larger size problem of choosing the optimal shot in the game of billiards. While not offering (and not expected to offer) any great new insights into the game, the method resulted in a model which is intuitively obvious and correct.

This work was supported by the Slovenian research agency ARRS (J2-2194, P2-0209).

Appendix

We need to prove that the ordering of probabilities $p(c|a_1,\ldots,a_i,\ldots,a_n)$ does not change if we omit from the conditional part the values which are conditionally independent from a_i given the class c. Let us first redefine the PDQ PD using conditional log odds ratios.

$$\frac{\partial_Q f}{\partial_Q A_i : a_i \to a_i'}(a_1,\ldots,a_n) = \\ \operatorname{sgn}(-\ln \frac{p(c|a_1,\ldots,a_i,\ldots,a_n)/p(\bar{c}|a_1,\ldots,a_i,\ldots,a_n)}{p(c|a_1,\ldots,a_i',\ldots,a_n)/p(\bar{c}|a_1,\ldots,a_i',\ldots,a_n)}), \tag{8}$$

where $\bar{c}$ is the complement of the target class c. It is easy to see that (8) is equivalent to (4).

Let us without loss of generality assume that values a_1 to a_k, $k < i$ are conditionally independent of values a_{k+1} to a_n, given the class. Applying Bayesian rule, using the independence assumption, cancelling the identical terms and reapplying the Bayesian rule turns (8) into

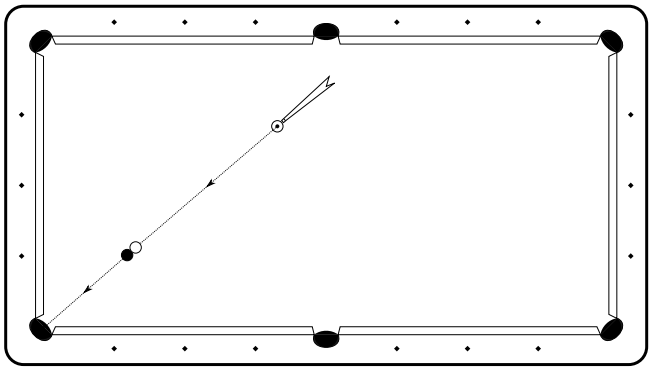
$$\frac{\partial_Q f}{\partial_Q A_i : a_i \to a_i'}(a_1,\ldots,a_n) = \\ \operatorname{sgn}(-\ln \frac{p(c|a_{k+1},\ldots,a_i,\ldots,a_n)/p(\bar{c}|a_{k+1},\ldots,a_i,\ldots,a_n)}{p(c|a_{k+1},\ldots,a_i',\ldots,a_n)/p(\bar{c}|a_{k+1},\ldots,a_i',\ldots,a_n)}). \tag{9}$$

This is equivalent to (4) without values a_1 to a_k. Therefore, $p(c|a_1,\ldots,a_i,\ldots,a_n) \leq p(c|a_1,\ldots,a_i',\ldots,a_n) \iff p(c|a_{k+1},\ldots,a_i,\ldots,a_n) \leq p(c|a_{k+1},\ldots,a_i',\ldots,a_n)$.

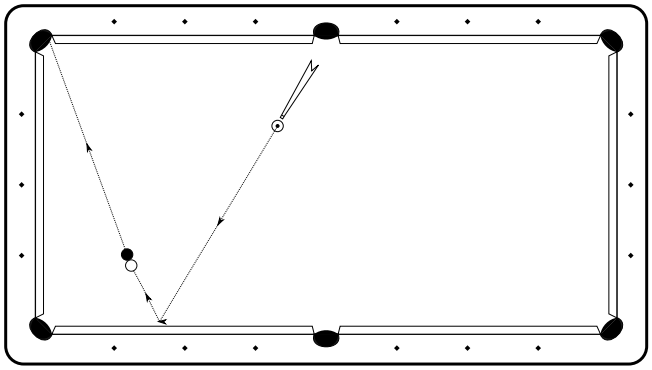
References

1. Alciatore, D.G.: The Illustrated Principles of Pool and Billiards. Sterling; 1st edition (August 2004)
2. Boutilier, C., Brafman, R.I., Hoos, H.H., Poole, D.: Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. Journal of Artificial Intelligence Research 21, 2004 (2003)
3. Brafman, R.I.: Preferences, planning and control. In: KR. pp. 2–5 (2008)
4. Bratko, I., Šuc, D.: Learning qualitative models. AI Magazine 24(4), 107–119 (2003)
5. Brochu, E., de Freitas, N., Ghosh, A.: Active preference learning with discrete choice data. In: Advances in Neural Information Processing Systems (2007)
6. Cestnik, B.: Estimating probabilities: A crucial task in machine learning. In: ECAI. pp. 147–149 (1990)
7. Chu, W., Ghahramani, Z.: Preference learning with gaussian processes. In: ICML '05: Proceedings of the 22nd international conference on Machine learning. pp. 137–144. ACM, New York, NY, USA (2005)
8. de Kleer, J., Brown, J.: A qualitative physics based on confluences. Artificial Intelligence 24, 7–83 (1984)
9. Demšar, J., Zupan, B., Leban, G., Curk, T.: Orange: From experimental machine learning to interactive data mining. In: PKDD. pp. 537–539 (2004)
10. Doyle, J.: Prospects for preferences. Computational Intelligence 20(2), 111–136 (2004)
11. Forbus, K.: Qualitative process theory. Artificial Intelligence 24, 85–168 (1984)
12. Forbus, K.: Qualitative reasoning. CRC Press (1997)
13. Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
14. Frank, E., Hall, M., Pfahringer, B.: Locally weighted naive Bayes. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI 2003) (2003)
15. Friedman, N., Goldszmidt, M.: Building classifiers using Bayesian networks. In: Proceedings of the thirteenth national conference on artificial intelligence. pp. 1277–1284. AAAI Press (1996)
16. Fürnkranz, J., Hüllermeier, E.: Preference learning. KI 19(1) (2005)
17. Klenk, M., Forbus, K.: Analogical model formulation for transfer learning in AP physics. Artificial Intelligence 173(18), 1615–1638 (2009)
18. Kononenko, I.: Semi-naive bayesian classifier. In: EWSL. pp. 206–219 (1991)
19. Kuipers, B.: Qualitative simulation. Artificial Intelligence 29, 289–338 (1986)
20. Kuipers, B.: Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge. MIT Press, Massachusetts (1994)
21. Langley, P., Sage, S.: Induction of selective Bayesian classifiers. In: Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence. pp. 399–406. Morgan Kaufmann (1994)
22. Papavasiliou, D.: Billiards manual. Tech. rep. (2009), http://www.nongnu.org/billiards/
23. Sokal, R.R., Michener, C.D.: A statistical method for evaluating systematic relationships. University of Kansas Scientific Bulletin 28, 1409–1438 (1958)
24. Žabkar, J., , Možina, M., Bratko, I., Demšar, J.: Learning qualitative relations from categorical data. In: Proc. of the 24th International Workshop on Qualitative Reasoning. Portland, USA (2010)
25. Žabkar, J., Bratko, I., Demšar, J.: Learning qualitative models through partial derivatives by Padé. In: Proc. of the 21th International Workshop on Qualitative Reasoning. Aberystwyth, U.K. (2007)
26. Žabkar, J., Možina, M., Bratko, I., Demšar, J.: Discovering monotone relations with padé. In: ECML 2009 workshop: Learning Monotone Models From Data (2009)

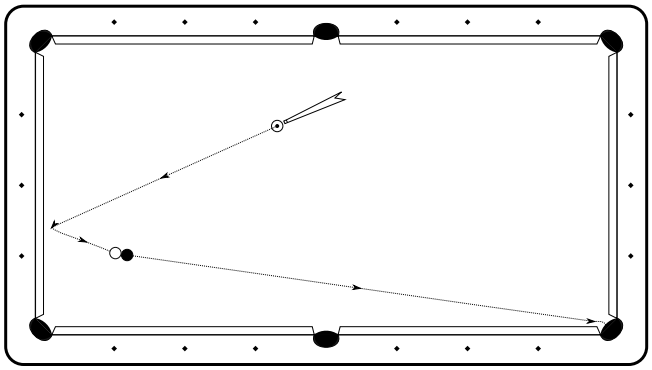
27. Zheng, Z., Webb, G.I., Ting, K.M.: Lazy Bayesian rules: a lazy semi-naive Bayesian learning technique competitive to boosting decision trees. In: Proc. 16th International Conf. on Machine Learning. pp. 493–502. Morgan Kaufmann, San Francisco, CA (1999)



(a) Direct shot.



(b) Left rail-first shot.



(c) Right rail-first shot.

Fig. 4: Examples of different type of shots used in our case study.