A compact fuzzy extension of the Naive Bayesian classification algorithm

Hans-Peter Störr[1]
AI Institute, Dept. Computer Science
Technische Universität Dresden
01062 Dresden, Germany
hans-peter@inf.tu-dresden.de

Abstract: We introduce a conservative fuzzy logic extension of the Naive Bayesian classification algorithm. The extension generalizes the algorithm such that the examples are described by a fuzzy set of attributes, instead of a classical set. Thus, an example possesses each attribute to a degree in $[0, 1]$. We present a new classification algorithm usable with fuzzy sets that is (a) fast, (b) is able to work with few training examples, (c) uses a compact representation of the internal model, (d) is able to deal with missing attributes, and (e) can be used for incremental learning, such that a rapid alternation of learning of new examples and classification of examples is possible. Our extension to Naive Bayesian classification is conservative in the sense that in the classical limit, when every fuzzy membership degree of the attributes is 0 or 1, our algorithm behaves exactly as the Naive Bayesian classification algorithm.

Key words: fuzzy sets, classification algorithms, Naive Bayesian classification

1. Introduction

Classification algorithms play an important role in numerous applications. There are many applications for sophisticated algorithms like C4.5 [12] or Bayesian Networks [6] that can handle large datasets and model complex classification functions. On the other hand, there are also “small” applications for classification algorithms in which speed and compact representation are more important than accuracy and ability to deal with large datasets.

Some of these “small” applications can be found in user modeling. Let us consider such an application. In [3] we describe a system implemented into e-shop systems of hybris [9], in which the products in the web shop are presented in a personalized way. The user can interactively change the presentation by enlarging / shrinking pictures, folding texts into headlines and vice versa, and so forth. Naturally, these interactions provide us with information on the users preferences, such that the system is able to construct a model of the preferences of each user. This model is then used to adapt the presentation of further web pages viewed by the user.

In our system, a web page is composed of content fragments that can be shown in a decollapsed resp. collapsed form, such as an large image resp. a thumbnail of that image, or a full text resp. only the headline of that text. These content fragments are automatically arranged in the web browser window to form an optically pleasant layout. When the user views a new web page, the page is adapted to her preferences by classifying its content fragments as interesting resp. not interesting for her, and showing in decollapsed resp. collapsed form.

The classification is based on attributes given to the fragments by the shop operator. For such an application we need a fast and compact classification algorithm, because the system has to solve classification tasks for each web page view of a potentially large number of visiting users based on relatively little data.

A (supervised) classification algorithm is an algorithm that takes a set of training examples as input. These are often described by attribute - value pairs, e.g., *type = picture* or *content = technical_details* for a content fragment. Usually, discrete or continuous attributes are distinguished, depending on whether the attributes can take a finite number of values or, for example, values in a range of real numbers. One of the discrete attributes is distinguished as the class attribute of the example. The classification algorithm uses these training examples to generate an internal model of the input data, with which it can predict the class of new, unseen, examples. (Some algorithms, e.g., instance based classificators, can do without an internal model, though.)

In [3] we discuss the adequacy of a number of compact learning algorithms in the discussed setting of an adaptive web presentation, namely CDL4 [13], ID3 [11], ITI [14], FID [10] and Naive Bayes classification [7]. Here, we feel that it is sensible to distinguish fuzzy attributes as a third kind of attribute. Consider for example an online-shop for clothing. The interest of the customer in different content fragments will usually be influenced by the product she/he is viewing. For instance, when buying fashionable clothes the customer will probably want to see pictures. On the other hand, when buying clothes where fashionability is not an issue, e.g., work clothes, the pictures might become less important in favor of technical details like fabric and manufacturing quality. But: what kind of an attribute is fashionability? Certainly, fashionability can be represented by a continuous (say, ranging from 0 to

[1]The author acknowledges support by hybris GmbH, Munich, Germany, within the project “Intelligent Systems in e-Commerce” (ISeC).

1) or a discrete attribute (say, ranging from *not at all* over *a little*, *more or less* to *very*). However, both approaches have shortcomings with many of the conventional classification algorithms. Different discrete values of attributes are often treated like they are completely unrelated (whereas the customer will probably judge informations about *not at all* fashionable products similar to informations about *a little* fashionable products). On the other hand, continuous attributes do reflect the relation between 0 (signifying *not at all* and 0.2 (signifying *a little*), but classification algorithms deal with them often by splitting the range into several sub-ranges that are considered separately. [2]In our case, however, this seems inappropriate, because there are usually only few training examples, making an elaborate division of the $[0, 1]$-range seem inappropriate. Thus, we opt for the use of fuzzy sets [15] of attributes, interpreting the value of the attribute as a membership degree that determines to which extent the example is described by the attribute. In effect, the values between 0 and 1 are treated with some form of "interpolation" of the extremes 0 and 1. We believe a degree seems a sensible kind of judgment wrt. a statement like "the product is fashionable".

Let us review the sketched main requirements for the classification algorithm in our application:

1. Fast, incremental learning. Since classification and adding new training examples take turns as the user alternates between interacting with a web page and calling new web pages, the classifier should not have to construct a new model from scratch every time new data is added.
2. Learning from few examples. The adaption of the presentation to the users preferences has to start immediately, because we usually cannot rely on data from data from the past - users often do not like to login.
3. A compact representation of the internal model.
4. As discussed, allow the use of fuzzy attributes.

In [8] we noted, that only few algorithms that satisfy these criteria are available, and developed an incremental algorithm IIFDT to construct and manipulate fuzzy decision trees. In this paper we present another new classification algorithm that satisfies the requirements above by extending the old and well–known Naive Bayesian classifier to fuzzy attributes. In spite of their simplicity and often unrealistic assumptions (see Section 2.), Naive Bayesian classifiers often yield surprisingly good results in comparison to decision tree or rule based systems [5, 4]. Furthermore, their results can sometimes be interpreted much more easily than large decision trees, and respect the quite "holistic" nature of human reasoning to some extent. Hence, they seem a good choice for our application, after we have paved the way to apply these to fuzzy attributes.

[2]A notable exception would be neural networks and related approaches, but most require a computationally expensive training phase making them less suitable for online applications.

While Naive Bayesian classification has been employed to continuous attributes in [2], the approach used there works only if one can give a probability distribution for the continuous attributes, which does not seem meaningful in the case of fuzzy truth values. So we employ another way to extend the algorithm to fulfill our requirements.

The remainder of the paper is organized as follows. In Section 2 we present the basic idea of the Naive Bayesian Classifier with discrete attributes. This is then extended to fuzzy attributes in Section 3. A short discussion of the handling of missing attributes is given in Section 4. Section 5 discusses complexity issues.

2. The Naive Bayesian Classifier

Let C denote a class attribute with a finite domain $\text{dom}(C)$ of m classes, and $V_1 \dots V_n$ a number attributes with finite domains $\text{dom}(V_1) \dots \text{dom}(V_n)$. An *example* e is described by its attribute values $v_1^e \in \text{dom}(V_1), \dots v_n^e \in \text{dom}(V_n)$. To simplify the presentation, we assume in the remainder of the paper that all of these values actually occur in the training examples, such that none of the discussed probabilities are zero. In practice, this demand can easily be fulfilled by only considering the values that already occured in the examples.

Naive Bayes classifiers implement a probabilistic idea of classification: they calculate the class of a new example e by estimating for each class from $\text{dom}(C)$ the probability that the example is in this class, and predict the most probable class as the class of e. Formally, for all $c \in \text{dom}(C)$ they estimate the probability

$$P(C = c \mid V_1 = v_1^e, V_2 = v_2^e, \dots, V_n = v_n^e)$$

that an example with attribute values like the given new example e has the class c. To improve readability, we will use in the following $P(\dots v_i^e \dots)$ as an abbreviation for $P(\dots V_i = v_i^e \dots)$, as well as $P(\dots c \dots)$ as an abbreviation for $P(\dots C = c \dots)$.

In practice, a complete table of these conditional probabilities would usually be too large to memorize, because its size is exponential in the number of attributes. Moreover, the number of training examples is often too small to give a good estimate for all these values. So, the idea of Naive Bayes classification is first to apply the Bayes rule

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)}$$

to the value $P(c \mid v_1^e \dots v_n^e)$ we are looking for:

$$P(c \mid v_1^e \dots v_n^e) = \frac{P(v_1^e \dots v_n^e \mid c)P(c)}{P(v_1^e \dots v_n^e)} \quad .$$

Since the table of all probabilities $P(v_1^e \dots v_n^e \mid c)$ is just as large as a table for $P(c \mid v_1^e \dots v_n^e)$, there is a second step: the “naive” assumption that all attribute values are independent given the class value

$$P(v_1^e \dots v_n^e \mid c) = P(v_1^e \mid c) \dots P(v_n^e \mid c) \ , \quad (1)$$
$$P(v_1^e \dots v_n^e) = P(v_1^e) \dots P(v_n^e) \ , \quad (2)$$

giving us the probability estimation

$$P(c \mid v_1^e \dots v_n^e) = \frac{P(v_1^e \mid c) \cdots P(v_n^e \mid c)}{P(v_1^e) \cdots P(v_n^e)} P(c) \ . \quad \text{(NBayes)}$$

The independence assumption is naive in that it is in general not met. Nevertheless, Naive Bayesian classifiers give quite good results in many cases, and are often a good way to perform classification when there is too little data on those dependencies to employ more sophisticated means. As [1] discusses, they can be seen as a special case of Bayesian Networks [6], which take attribute dependencies into consideration.

In an application using (NBayes) for classification of examples, $P(c \mid v_1^e \dots v_n^e)$ is calculated for all $c \in \text{dom}(C)$ to find the maximum value. Here, the denominator in (NBayes) serves as a normalizing factor that can be omitted from the calculation.

A further issue is the estimation of the probability values $P(v_i^e \mid c)$ and $P(v_i^e)$ given a set $\mathcal{L}$ of training examples. The “obvious” solution

$$\begin{aligned} P(v_i^e) &\approx \frac{\mid \mathcal{L}|_{V_i = v_i^e} \mid}{\mid \mathcal{L} \mid} \\ P(c) &\approx \frac{\mid \mathcal{L}|_{C=c} \mid}{\mid \mathcal{L} \mid} \\ P(v_i^e \mid c) &\approx \frac{\mid \mathcal{L}|_{V_i = v_i^e, C=c} \mid}{\mid \mathcal{L}|_{C=c} \mid} \end{aligned} \quad (3)$$

is not well suited for the estimation of these probabilities from small sets $\mathcal{L}$, because it often estimates the probabilities to extreme values like 1 or 0. If, for instance, a class value does not occur in a training set of 4 examples, it seems warranted to estimate its probability to, say, a value in the interval $[0, \frac{1}{4}]$, but the value 0 calculated by (3) is the very edge of that interval, and thus seems a rather bad estimate. The Laplace-correction smoothes the calculated probability values: when an experiment with k possible outcomes is performed n times, the probability of an outcome occurring r times is estimated to $\frac{r+1}{n+k}$. Applied to our setting, this yields the estimations[3]

$$\begin{aligned} P(v_i^e) &= \frac{\mid \mathcal{L}|_{V_i = v_i^e} \mid +1}{\mid \mathcal{L} \mid + \mid \text{dom}(V_i) \mid} \ , \\ P(c) &= \frac{\mid \mathcal{L}|_{C=c} \mid +1}{\mid \mathcal{L} \mid + \mid \text{dom}(C) \mid} \ , \\ P(v_i^e \mid c) &= \frac{\mid \mathcal{L}|_{V_i = v_i^e, C=c} \mid +1}{\mid \mathcal{L}|_{C=c} \mid + \mid \text{dom}(V_i) \mid} \ . \end{aligned} \quad (4)$$

[3]A pleasant side effect of using the Laplace correction is that we do not have to deal with zero probabilities in the denominator of (NBayes).

Summing up: Naive Bayes classification for discrete attribute values is performed for an example e by predicting its class to the value $c \in \text{dom}(C)$ with maximum probability $P(c \mid v_1^e \dots v_n^e)$ calculated as in (NBayes) using estimations for the needed probabilities as, e.g., in (4).

3. The fuzzy extension

In the fuzzy case we generalize the concept of an attribute such that an example e does not have exactly one value $v_i^e \in \text{dom}(V_i)$ for each attribute V_i, but has each value $v_i \in \text{dom}(V_i)$ to a degree $\mu_{v_i}^e \in [0, 1]$. This corresponds to the concept of a linguistic variable in fuzzy logic: the attribute names a linguistic variable, and each of the values of this attribute corresponds to a linguistic term. The restriction to one class per training example is also lifted: they are classified to a degree $\mu_c^e \in [0, 1]$ for each class $c \in \text{dom}(C)$. We assume these degrees are normalized for each example:

$$\sum_{v_i \in \text{dom}(V_i)} \mu_{v_i}^e = 1 \qquad \sum_{c \in \text{dom}(C)} \mu_c^e = 1 \quad (5)$$

For the moment, let us reinterpret these degrees probabilistically. Let us think of the degree $\mu_{v_i}^e$ as the probability $P(v_i \mid e)$ an example e has an attribute v_i. This is not the intrinsic meaning of a fuzzy truth degree - a product is not fashionable with a probability 30% but rather to a degree 0.3, but we will see it allows us to extend the Naive Bayes classification in a natural way.

$$P(v_i^e = v_i \mid e) = \mu_{v_i}^e \qquad P(c_e = c \mid e) = \mu_c^e \quad (6)$$

To avoid cluttering, we use $P(\dots v_i \dots \mid e)$ as an abbreviation for $P(\dots v_i^e = v_i \dots \mid e)$. Furthermore, we assume all attribute values and the class value are independent in this artificial probability distribution for e. So, we can calculate the probability $P(c \mid e)$ an example e has class c as follows. First, we split over all cases for the actual attribute values, assuming the class depends only on these values.

$$P(c \mid e) = \sum_{v_1 \in V_1, \dots v_n \in V_n} P(c \mid v_1 \dots v_n) P(v_1 \dots v_n \mid e)$$

Next, we make the assumption that the attribute values of example e are independent of each other. Note that these probabilities are subject of our reinterpretation of the fuzzy truth degrees. We obtain:

$$\sum_{v_1 \in V_1, \dots v_n \in V_n} P(c \mid v_1 \dots v_n) P(v_1 \mid e) \cdots P(v_n \mid e)$$

At this point, we can lift our reinterpretation of the fuzzy truth values and go back to membership degrees using (6). So we get:

$$\sum_{v_1 \in V_1, \dots v_n \in V_n} P(c \mid v_1 \dots v_n) \mu_{v_1}^e \cdots \mu_{v_n}^e$$

Application of the Bayes rule to $P(c \mid v_1 \ldots v_n)$ yields:

$$\sum_{v_1 \in V_1, \ldots v_n \in V_n} \frac{P(v_1 \ldots v_n \mid c) P(c)}{P(v_1 \ldots v_n)} \mu^e_{v_1} \cdots \mu^e_{v_n}$$

Now we apply the same naive independence assumption as in the classical case.

$$\sum_{v_1 \in V_1, \ldots v_n \in V_n} \frac{P(v_1 \mid c) \cdots P(v_n \mid c) P(c)}{P(v_1) \cdots P(v_n)} \mu^e_{v_1} \cdots \mu^e_{v_n}$$

We move constant factors in front of the first sum

$$P(c) \sum_{v_1 \in V_1} \frac{P(v_1 \mid c)}{P(v_1)} \mu^e_{v_1} \sum_{v_2 \in V_2, \ldots v_n \in V_n} \frac{P(v_2 \mid c) \cdots P(v_n \mid c)}{P(v_2) \cdots P(v_n)} \mu^e_{v_2} \cdots \mu^e_{v_n} \quad ,$$

and repeat this with the other sums. Finally, we find using distributivity:

$$P(c \mid e) = P(c) \left(\sum_{v_1 \in V_1} \frac{P(v_1 \mid c)}{P(v_1)} \mu^e_{v_1} \right) \cdots \left(\sum_{v_n \in V_n} \frac{P(v_n \mid c)}{P(v_n)} \mu^e_{v_n} \right) \quad \text{(FNBayes)}$$

Observe that this formula is quite similar to the classical case (NBayes). Indeed, if we consider the extreme case when for each attribute V_i only one of the $\mu^e_{v_i}$ with $v_i \in \text{dom}(V_i)$ is 1 and the others are 0, then the example is described as in the classical case by the attribute values for which $\mu^e_{v_i} = 1$, and we get the same result with both formulas. Varying the truth degrees between those extremes yields a continuous transition between the classically calculated values. Note, that we can apply (FNBayes) both to classify the example to a "crisp" class value by returning the class with maximum probability, or treat the probabilities like fuzzy truth values, returning them as truth degrees μ^e_c for all classes $c \in \text{dom}(C)$.

To apply (FNBayes) to our classification task, we have to estimate the probabilities used in (FNBayes). We can carry over the spirit of (4) to our fuzzy setting by using the corresponding fuzzy set sizes. I.e., to find the size of a set we calculate the sum of the degrees the examples belong to that set. For instance, $\mid L|_c \mid$ becomes $\sum_{e \in \mathcal{L}} \mu^e_c$. [4] Again, it seems advisable to use the Laplace correction for small training sets. So we find:

$$P(v_i) = \frac{\left(\sum_{e \in \mathcal{L}} \mu^e_{v_i} \right) + 1}{\mid \mathcal{L} \mid + \mid \text{dom}(V_i) \mid}$$

$$P(c) = \frac{\left(\sum_{e \in \mathcal{L}} \mu^e_c \right) + 1}{\mid \mathcal{L} \mid + \mid \text{dom}(C) \mid} \tag{7}$$

$$P(v_i \mid c) = \frac{\left(\sum_{e \in \mathcal{L}} \mu^e_{v_i} \mu^e_c \right) + 1}{\left(\sum_{e \in \mathcal{L}} \mu^e_c \right) + \mid \text{dom}(V_i) \mid} \quad .$$

4. Missing Attributes

In many practical cases it happens that some examples have missing attributes, i.e., for some of the attributes the value for that example is not known. One can imagine a number of ways how to handle this problem. The best way to do this depends on the application.

Let us first consider the case that these attributes are missing due to some kind of "transmission failure". That is: in principle there are values for these attributes of the examples, but we just don't know them. Both in the case of Naive Bayes classification and our fuzzy variant, we can easily classify examples with missing attributes by simply omitting these attributes from the calculation. If, on the other hand, attributes are missing from examples of the training set, we have to modify the equations (4) resp. (7) for estimating the probabilities such that only the part $\mathcal{L}_{V_i}$ of the training set $\mathcal{L}$, for which the considered attribute V_i is not missing, is considered. For instance, we would have

$$P(v_i) = \frac{\left(\sum_{e \in \mathcal{L}_{V_i}} \mu^e_{v_i} \right) + 1}{\mid \mathcal{L}_{V_i} \mid + \mid \text{dom}(V_i) \mid} \quad .$$

In some applications, however, missing attributes occur because the attributes simply do not apply to some of the examples. A clothing shop, for instance, might sell miscellaneous items like coat hangers, for which an attribute like fashionability has no meaning. If we just ignore the attribute like suggested above, we lose something because the fact of the absence of this attribute does convey information about the example. So we might be better off by introducing an additional value *absent* in the domain of such attributes, that replaces a missing value in an example.

5. Complexity Considerations

Let us consider the space and time complexities of the algorithms. To use Naive Bayesian classification resp. its fuzzy extension in an application, we have

[4]This is consistent with our temporary probabilistic reinterpretation: what we get this way is the mean size of the set $\mid L|_c \mid$, assuming $P(c \mid e) = \mu^e_c$. The alert reader will notice that we implicitly use the algebraic t-norm in (7), again consistent with our reinterpretation of fuzzy degrees of truth.

to implement two basic operations: adding a new training example to $\mathcal{L}$, and classification of an example. In the following we discuss the time complexities of these operations, as well as the space needed for the database used in the operations. As the reader can easily verify, the additional space needed for performing both mentioned operations is less than the space needed for the database, and is therefore not an issue.

Let $k = |\text{dom}(C)|$ be the number of classes and $v = \max_{i=1...n} |\text{dom}(V_i)|$ the maximal number of values of an attribute. To apply the Naive Bayesian classifier we have to memorize and update the values $|\,\mathcal{L}\,|$, $|\,\mathcal{L}|_{C=c}\,|$, $|\,\mathcal{L}|_{V_i=v_i}\,|$, and $|\,\mathcal{L}|_{V_i=v_i,C=c}\,|$ for all $c \in \text{dom}(C)$, $i = 1 \dots n$ and $v_i \in \text{dom}(V_i)$, which are used with (4) and (NBayes) for the classification of new examples. This gives a space complexity of $O(knv)$ for the database, as well as a time complexity of $O(n)$ for updating these values with data from a new example. For the classification of an example we get the time complexity $O(kn)$, because (NBayes) has to be calculated for all k classes $c \in \text{dom}(C)$.

For the fuzzy extension of the Naive Bayesian classifier, the database consists of the values used at the right sides of (7): $|\,\mathcal{L}\,|$, $\sum_{e\in\mathcal{L}} \mu^e_{v_i}$, $\sum_{e\in\mathcal{L}} \mu^e_c$, $\sum_{e\in\mathcal{L}} \mu^e_c$, and $\sum_{e\in\mathcal{L}} \mu^e_{v_i}\mu^e_c$ for all $c \in \text{dom}(C)$, $i = 1 \dots n$ and $v_i \in \text{dom}(V_i)$. So, again, the size of the database is $O(knv)$. Since an example in the fuzzy extension can have all attribute and class values to a nonzero degree, updating the database with a new example can require $O(knv)$ steps. The classification of an example requires the k-fold computation of (FNBayes), and thus $O(knv)$ steps.

Table 1 summarizes the complexities. Note that all values are independent from the number of learned examples $|\,\mathcal{L}\,|$ - the algorithm does not need to memorize the examples, but rather keeps "bookkeeping values" in its database, which summarize the examples. Therefore, we think the algorithm is suited very well for incremental application.

In many applications, like our discussed application, k and v are small values like 2 or 3; only n is large. In this case, the algorithm complexity is optimal in a sense that the database space and the runtimes of the adding and classification procedures are in the order of the size of the examples added / classified – one can only fall significantly below this value when parts of the example are not used at all.[5] Thus, we consider the algorithm as fast and having a compact internal representation.

6. Conclusion

We have introduced an extension of the supervised Naive Bayesian classification algorithm that allows the use of fuzzy attributes. We argue that this algorithm is suitable to work in applications that provide only small training sets and require a compact representation of the internal model. The algorithm is easily able to work in an incremental manner because new examples just require an update of the sums and set sizes used at the right side of (7). (In contrast, for instance the incremental variants of decision tree algorithms like ITI and IIFDT usually require the storage of all examples used while constructing the tree, and periodically require an computationally expensive restructuring of the decision trees.) We feel that the algorithm can be a valuable alternative in the set of existing classification algorithms.

[5]Some algorithms, like decision tree based algorithms, do indeed only use parts of the example for the classification. The price payed are more expensive computations to find out which parts of the example are most important. Thus, the "add" operation is more expensive.

	N.Bayes	Fuzzy N.Bayes
example size	$O(n)$	$O(k+nv)$
d.b. space	$O(knv)$	$O(knv)$
add	$O(n)$	$O(knv)$
classify	$O(kn)$	$O(knv)$

Table 1: Space for one example, space of the database and time complexities for the operations adding an example and classifying an example for the discussed algorithms.
$k = |\text{dom}(C)|$ and $v = \max_{i=1...n} |\text{dom}(V_i)|$.

References

[1] Christian Borgelt, Heiko Timm, and Rudolf Kruse. Using fuzzy clustering to improve naive bayes classifiers and probabilistic networks. In *Proc. 8th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, San Antonio, TX, USA, 2000. IEEE Press, Piscataway, NJ, USA.

[2] Christian Borgelt, Heiko Timm, and Rudolf Kruse. Probabilistic networks and fuzzy clustering as generalizations of naive bayes classifiers. In Bernd Reusch and Karl-Heinz Temme, editors, *Computational Intelligence in Theory and Practice*, Advances in Soft Computing, pages 121–138. Physica-Verlag, Heidelberg, Germany, 2001.

[3] Sven-Erik Bornscheuer, Yvonne McIntyre, Steffen Hölldobler, and Hans-Peter Störr. User adaptation in a web shop system. In M. H. Hamza, editor, *Proceedings of the IASTED International Conference Internet and Multimedia Systems and Applications*, pages 208–213, Anaheim, Calgary, Zurich, 2001. ACTA Press.

[4] Peter Clark and Tim Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

[5] Pedro Domingos and Michael Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 105–112, Bari, Italy, 1996. Morgan Kaufmann.

[6] N. Friedman and M. Goldszmith. Building classifiers using bayesian networks. In *Proceedings of the thirteenth National Conference on Artificial Intelligence*, pages 1277–1284, Menlo Park, CA, 1996. AAAI Press.

[7] I. J. Goof. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. MIT Press, Cambridge, CA, USA, 1965.

[8] Marina Guetova, Steffen Hölldobler, and Hans-Peter Störr. Incremental fuzzy decision trees. In *Proceedings of the 25th German Conference on Artificial Intelligence (KI2002)*, Aachen, Germany, 2002.

[9] hybris GmbH. Hybris jakarta. `http://www.hybris.de/`, 2002.

[10] Cezary Z. Janikow. Fuzzy decision trees: Issues and methods. *IEEE Transactions on Systems, Man, and Cybernetics*, 28(1):1–14, 1998.

[11] J. R. Quinlan. Induction on decision trees. *Machine Learning*, 1:81–106, 1986.

[12] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[13] Wei-Min Shen. An active and semi-incremental algorithm for learning decision lists. Technical Report USC-ISI-97, Information Sciences Institute, University of Southern California, 1997.

[14] Paul E. Utgoff, Neil C. Berkman, and Jeffery A. Clouse. Decision tree induction based on efficient tree restructuring. *Machine Learning*, 29:5–44, 1997.

[15] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.