Using Fuzzy Clustering to Improve Naive Bayes Classifiers and Probabilistic Networks

Christian Borgelt, Heiko Timm, and Rudolf Kruse
Dept. of Knowledge Processing and Language Engineering
Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany

Abstract— **Although probabilistic networks and fuzzy clustering may seem to be disparate areas of research, they can both be seen as generalizations of naive Bayes classifiers. If all descriptive attributes are numeric, naive Bayes classifiers often assume an axis-parallel multidimensional normal distribution for each class. Probabilistic networks remove the requirement that the distributions must be axis-parallel by taking covariances into account where this is necessary. Fuzzy clustering tries to find general or axis-parallel distributions to cluster the data. Although it neglects the class information, it can be used to improve the result of the abovementioned methods by removing the restriction to only one distribution per class.**

I. Introduction

Probabilistic networks are a method to decompose a multivariate probability distribution in order to make reasoning in multidimensional domains feasible. Fuzzy clustering is a method to find groups of similar objects or cases, which compared to classical (crisp) clustering has the advantage that an object or a case can belong (with a degree between 0 and 1) to more than one cluster. Hence these methods may appear to be relatively unrelated. Nevertheless, they are closely connected, since they can both be seen as generalizations of naive Bayes classifiers.

Our rationale is that the three techniques share the idea that the process that generated a given dataset can be modeled by a set of probability distributions/density functions. They differ w.r.t. the assumptions they make about these distributions and whether they take into account the value of a distinguished class attribute (supervised: naive Bayes classifiers, probabilistic networks) or not (unsupervised: fuzzy clustering). Of course, there are still other methods, for example, radial basis function neural networks [18], that can be interpreted in much the same way. However, a complete list of such methods and a discussion of their similarities and differences is beyond the scope of this paper. We selected the three methods mentioned above as examples, because the first two show very clearly the properties we are interested in and because the connection to fuzzy clustering points out interesting directions to improve these techniques.

To simplify the explanation, we confine ourselves to numeric attributes (with the exception of the class attribute, of course). With this restriction a common assumption is that the process that generated the data can be modeled by a set of multidimensional normal distributions. The three methods differ in the constraints they place on this set. Naive Bayes classifiers and probabilistic networks assume exactly one distribution per class. Naive Bayes classifiers, in addition, assume that the descriptive attributes are independent given the class, thus requiring the distributions to be axis-parallel. Analogously, there are general and axis-parallel variants of fuzzy clustering algorithms. In fuzzy clustering, however, the number of distributions can be chosen freely. This often leads to a better fit to the data and may be used to improve the two other methods.

The brief overview just given already fixes the order in which we discuss the methods. In section II we examine naive Bayes classifiers. In section III we show how Bayesian networks remove the strong independence assumptions underlying naive Bayes classifiers. In section IV we show how fuzzy clustering approaches can be used to improve the aforementioned methods by removing the restriction to one distribution per class.

II. Naive Bayes Classifiers

Naive Bayes classifiers [10], [5], [15], [16] are an old and well-known type of classifiers, i.e., of programs that assign a class from a predefined set to an object or case under consideration based on the values of descriptive attributes. They do so using a probabilistic approach, i.e., they try to compute conditional class probabilities and then predict the most probable class. Let C denote a class attribute with a finite domain of m classes, $\mathrm{dom}(C) = \{c_1, \ldots, c_m\}$, and let $U = \{A_1, \ldots, A_n\}$ be a set of descriptive attributes. These attributes may be symbolic, $\mathrm{dom}(A_j) = \{a_1^{(j)}, \ldots, a_{m_j}^{(j)}\}$, or numeric, $\mathrm{dom}(A_j) = \mathbb{R}$. For simplicity, we use the notation $a_{i_j}^{(j)}$ for both symbolic and numeric values. With this notation, a case or an object can be described by an instantiation $\omega = (a_{i_1}^{(1)}, \ldots, a_{i_n}^{(n)})$ of the attributes $A_1, \ldots, A_n$ and the universe of discourse is $\Omega = \mathrm{dom}(A_1) \times \ldots \times \mathrm{dom}(A_n)$.

For a given instantiation ω, a naive Bayes classifier tries to compute the conditional probability

$$P(C = c_i \mid \omega) = P\left(C = c_i \,\middle|\, \bigwedge\nolimits_{j=1}^{n} A_j = a_{i_j}^{(j)}\right)$$

for all c_i and then predicts the class c_i for which this probability is highest. Since it is usually impossible to store all of these conditional probabilities explicitly, naive Bayes classifiers exploit—as their name already indicates—Bayes rule and a set of conditional independence assumptions. With Bayes rule we get [1]

$$P\left(C = c_i \,\middle|\, \bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right) = \frac{f\left(\bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)} \,\middle|\, C = c_i\right) \cdot P(C = c_i)}{f\left(\bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right)},$$

assuming that always $f\big(\bigwedge_{A_j \in U} A_j = a_{i_j}^{(j)}\big) > 0$. It is clear that we can neglect the denominator of the fraction on the right, since for a given case or object to be classified, it is fixed and therefore can always be restored from the fact that the distribution on the classes must be normalized.

Next we apply the chain rule of probability to get

$$P\left(C = c_i \,\middle|\, \bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right) = \frac{1}{S} \cdot P(C = c_i) \cdot \prod_{k=1}^{n} f\left(A_k = a_{i_k}^{(k)} \,\middle|\, \bigwedge_{j=1}^{k-1} A_j = a_{i_j}^{(j)}, C = c_i\right),$$

where S is a normalization constant that represents the denominator of the fraction in the preceding equation. Finally we make the truly "naive" assumption that given the value of the class attribute, any attribute A_j is independent of any other. This yields

$$P\left(C = c_i \,\middle|\, \bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right) = \frac{1}{S} \cdot P(C = c_i) \cdot \prod_{j=1}^{n} f\left(A_j = a_{i_j}^{(j)} \,\middle|\, C = c_i\right),$$

the fundamental formula underlying naive Bayes classifiers. For a symbolic attribute A_j the conditional probabilities $P(A_j = a_{i_j}^{(j)} \mid C = c_i)$ are stored as a simple conditional probability table. For numeric attributes it is usually assumed that the probability density is a normal distribution and hence only the expected values $\mu_j(c_i)$ and the variances $\sigma_j^2(c_i)$ need to be stored. Alternatively, numeric attributes may be discretized [4] and then treated like symbolic attributes. In this paper, however, we make the normal distribution assumption. It is obvious that naive Bayes classifiers can easily be induced from from a dataset of preclassified sample cases using a standard estimation method like, e.g., maximum likelihood.

III. Probabilistic Networks

Probabilistic networks—especially Bayesian networks [19] and Markov networks [17]—are well-known tools for reasoning under uncertainty. They exploit independence relations between attributes in order to decompose a multivariate probability distribution into a set of (conditional or marginal) distributions on lower-dimensional subspaces. Early efficient implementations include HUGIN [1] and PATHFINDER [11].

[1]For simplicity, we always use a density function f, although this is strictly correct only, if there is at least one numeric attribute.

Dependence and independence relations have been studied extensively in the field of graphical modeling [13], [23]. Though using graphical models to facilitate reasoning in multidimensional domains has originated in the probabilistic setting, this approach has been generalized to be usable with other uncertainty calculi [21], e.g., in the so-called valuation-based networks [22] and has been implemented, for example, in PULCINELLA [20]. Due to their connection to fuzzy systems recently possibilistic networks also gained some attention. They have been implemented, for example, in POSSINFER [9], [14]. In this paper, however, we focus on Bayesian networks.

A Bayesian network is a directed acyclic graph in which each node represents an attribute and each edge represents a direct dependence between two attributes. The structure of the directed graph encodes a set of conditional independence statements. It also represents a particular joint probability distribution, which is specified by assigning to each node in the network a family of (conditional) probability distribution for the values of the corresponding attribute given its parents in the network (if any).

Formally, a Bayesian network describes a factorization of a multivariate probability distribution. This factorization results from applying first the chain rule of probability to a joint distribution, which yields

$$f\left(\bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right) = \prod_{j=1}^{n} f\left(A_j = a_{i_j}^{(j)} \,\middle|\, \bigwedge_{k=1}^{j-1} A_k = a_{i_k}^{(k)}\right).$$

Then the factors are simplified by exploiting *conditional independence statements* of the form

$$\forall \omega \in \Omega : P(\omega_{X \cup Y} \mid \omega_Z) = P(\omega_X \mid \omega_Z) \cdot P(\omega_Y \mid \omega_Z)$$

whenever $P(\omega_Z) > 0$, where X, Y, and Z are three disjoint sets of attributes and $\omega_X = \mathrm{proj}_X(\omega)$ is the projection of an instantiation $\omega = (\bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)})$ to the attributes in X. Since these statements are equivalent to

$$\forall \omega \in \Omega : P(\omega_X \mid \omega_{Y \cup Z}) = P(\omega_X \mid \omega_Z)$$

we can remove those attributes from the conditions of which the conditioned attribute is independent given the values of the remaining attributes. Thus we arrive at

$$\forall \omega \in \Omega : f(\omega) = f\left(\bigwedge_{j=1}^{n} A_j = a_{i_j}^{(j)}\right) = \prod_{j=1}^{n} f\left(A_j = a_{i_j}^{(j)} \,\middle|\, \omega_{\mathrm{parents}(A_j)}\right),$$

where parents(A_j) is the set of attributes of which to know the instantiations is sufficient to determine the probability (density) of the values of attribute A_j. The name "parents(A_j)" is due to the fact that in a Bayesian network the conditioning attributes are the parents of this

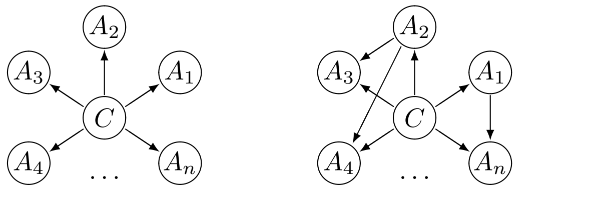

Fig. 1. A naive Bayes classifier is a Bayesian network with a star-like structure (left). It can be extended by adding edges between attributes that are still dependent given the class (right).

attribute in the graph. This makes it very simple to read the factorization from a Bayesian network: For each attribute there is one factor in which it is conditioned on the attributes corresponding to its parents in the graph.

Clearly, a sparse graph is desirable to obtain a factorization with "small" factors. Whether a sparse graph can be found sometimes depends on the order of the attributes, but it cannot be guaranteed that a sparse graph exists. In such cases usually an approximation is accepted.

Bayesian networks can be used for probabilistic reasoning by fixing the values of some (observed) attributes and then propagating this information in the network to obtain the probabilities of the values of other (unobserved) attributes. This process, which is usually called *evidence propagation*, basically consists in replacing the prior probability distribution with the posterior one, i.e., the one conditioned on the values of the observed attributes. To make it efficient, a Bayesian network is often transformed into a clique tree for which a simple propagation scheme exists. The evidence is propagated along the edges of this clique tree using the marginal probability distributions associated with the nodes that represent the cliques. For details on this method, see e.g. [17].

It is easy to see that a naive Bayes classifier is just a special Bayesian network with a star-like structure as shown on the left in figure 1. That is, there is a distinguished attribute, namely the class attribute. It is the only unconditioned attribute (the only one without parents). All other attributes are conditioned on the class attribute and on the class attribute only.

The main drawback of naive Bayes classifiers are the very strong conditional independence assumptions underlying them. By exploiting the more general approach underlying Bayesian networks, this severe constraint can be relaxed. That is, we may add edges between attributes that are still dependent given the class (see figure 1). This can lead to improved classification results, since the extended conditional probability distributions may be better suited to capture the dependence structure. To keep the resulting graph sparse, one may introduce the restriction that no attribute may have more than a fixed number of parents. Probabilistic networks of this type have been successfully applied in telecommunication [7].

As an illustrative example, let us take a look at the well-known iris data. The problem consists in predicting the iris type (iris setosa, iris versicolor, or iris virginica) from measurements of the sepal length and width and the petal length and width. We confine ourselves to the latter two measures, which are most informative. The naive Bayes classifier induced from all 150 cases is shown in table I. The conditional probability density functions used by this classifier are depicted in figure 2: The ellipses are the 2σ-boundaries of the (bivariate) normal distributions. Due to the strong conditional independence assumptions, these ellipses are axis-parallel (no covariance is taken into account). However, even a superficial glance reveals that the two measures are far from independent given the iris type. If we allow for an additional edge between the petal length and width, which is easily implemented by estimating the covariance of the two measures, a much better fit to the data can be achieved (see figure 3, again the el-

TABLE I

A NAIVE BAYES CLASSIFIER FOR THE IRIS DATA. THE NORMAL DISTRIBUTIONS ARE DESCRIBED BY $\hat{\mu} \pm \hat{\sigma}$.

iris type	iris setosa	iris versicolor	iris virginica
prior prob.	0.333	0.333	0.333
petal length	1.46 ± 0.17	4.26 ± 0.46	5.55 ± 0.55
petal width	0.24 ± 0.11	1.33 ± 0.20	2.03 ± 0.27

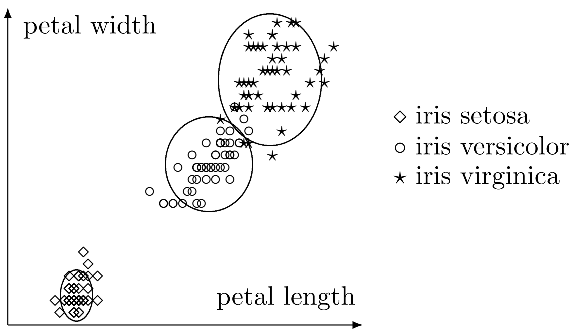

Fig. 2. Naive Bayes density functions for the iris data. The ellipses are the 2σ-boundaries.

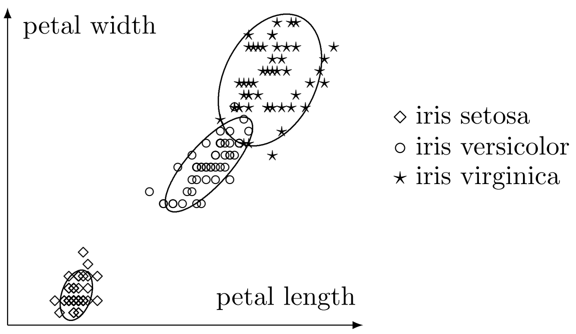

Fig. 3. Density functions that take into account the covariance of the two measures. The ellipses are the 2σ-boundaries.

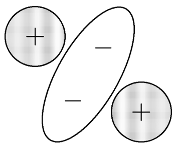

Fig. 4. An extreme 2-class example where an approach with one distribution per class fails.

lipses are the 2σ-boundaries of the density functions). As a consequence the number of misclassifications drops from six to three (which can easily be made out in figure 3).

To summarize, probabilistic networks generalize naive Bayes classifiers in two ways. In the first place, by additional edges, conditional dependences between attributes can be taken into account. Secondly, in probabilistic networks there is usually no distinguished class attribute. Any attribute (or any set of attributes) can be made the focus of inferences. Thus several, quite different reasoning tasks can be solved with the same probabilistic network.

However, there is still the restriction to one density function per class. This is not always appropriate, especially under the normal distribution assumption. An extreme example is shown in figure 4. It is clear that for this classification problem an approach with only one density function per class fails, although two density functions for the +-class can fit the data perfectly. To arrive at such a more general approach, we may exploit ideas from fuzzy clustering, which we study next.

IV. Fuzzy Clustering

The term "classification" is ambiguous. Up to now we have used it to denote the process of assigning a class from a *predefined* set. In classical statistics, however, this term usually denotes the process of dividing a dataset of sample cases, with the groups *not* predefined, but to be found by the algorithm. To avoid confusion, the latter process is often called *clustering* or *cluster analysis.*

Cluster analysis is, as already mentioned, a technique to divide a dataset of sample cases into classes or *clusters.* The goal is to divide the dataset in such a way that two cases from the same cluster are as similar as possible and two cases from different clusters are as dissimilar as possible. Thus one tries to imitate the human ability to group similar objects or cases into classes and categories.

In classical cluster analysis [6] each case or object is assigned to exactly one cluster. Thus we get a crisp partitioning with "sharp" boundaries between the clusters. A crisp partitioning of a dataset, however, though often undisputedly successful, is not always appropriate. If the "clouds" formed by the data points are not clearly separated by regions bare of any data points, but if, in contrast, there are only regions of higher and lesser data point density, then the boundaries between the clusters can only be drawn with a certain amount of arbitrariness. Due to this arbitrariness it may be doubted, at least for data points close to the boundaries, whether a definite assignment to one class is justified.

An intuitive approach to deal with such situations is to make it possible that a data point belongs in part to one cluster, in part to a second etc. *Fuzzy cluster analysis* does just this: It relaxes the requirement that a data point must be assigned to exactly one cluster by allowing gradual memberships [2], [3].

Most fuzzy clustering algorithms determine an optimal classification by minimizing an objective function. Each cluster is represented by a *cluster prototype.* This prototype consists of a *cluster center* and maybe some additional information about the size and the shape of the cluster. The cluster center is an instantiation of the attributes used to describe the domain (which may or may not appear in the dataset). The size and shape parameters determine the extension of the cluster in different directions of the underlying domain.

The degree of membership to which a given data point belongs to a given clusters is computed from the distance of the data point to the cluster center w.r.t. the size and the shape of the cluster. The closer it is to the center, the higher is its degree of membership. Hence the problem to divide a dataset $X = \{\vec{x}_1, \ldots, \vec{x}_r\} \subseteq \mathbb{R}^n$ into m clusters consists in minimizing the distances of the data points to the cluster centers, since we want to maximize the degrees of membership. That is, we have to minimize

$$J(\mathbf{X}, \mathbf{U}, \mathbf{B}) = \sum_{i=1}^{m} \sum_{j=1}^{r} u_{ij}^{\alpha} d^2(\vec{\beta}_i, \vec{x}_j) \qquad (1)$$

subject to

$$\sum_{j=1}^{r} u_{ij} > 0, \qquad \text{for all } i \in \{1, \ldots, m\}, \qquad (2)$$

$$\sum_{i=1}^{m} u_{ij} = 1, \qquad \text{for all } j \in \{1, \ldots, r\}, \qquad (3)$$

where $u_{ij} \in [0,1]$ is the membership degree of datum $\vec{x}_j$ to cluster c_i, $\vec{\beta}_i$ is the prototype of cluster c_i, and $d(\vec{\beta}_i, \vec{x}_j)$ is the distance between datum $\vec{x}_j$ and prototype $\vec{\beta}_i$. $\mathbf{B}$ is the set of all m cluster prototypes $\vec{\beta}_1, \ldots, \vec{\beta}_m$. The $m \times r$ matrix $\mathbf{U} = [u_{ij}]$ is called the fuzzy partition matrix and the parameter α is called the *fuzzifier.* This parameter determines the "fuzziness" of the classification. Higher values make the boundaries between the clusters softer, lower values make them harder. Usually $\alpha = 2$ is chosen. Constraint (2) guarantees that no cluster is empty and constraint (3) ensures that the sum of the membership degrees equals 1 for each datum. Fuzzy clustering algorithms which minimize the objective function J subject to these constraints are usually called *probabilistic clustering algorithms*, since the membership degrees for a given datum formally resemble the probabilities of its being a member of the different clusters.

The objective function $J(\mathbf{X}, \mathbf{U}, \mathbf{B})$ is usually minimized by updating the membership degrees u_{ij} and the prototypes $\vec{\beta}_i$ in an alternating fashion, until the change $\Delta\mathbf{U}$ of the membership degrees is less than a given bound ε.

The update formulae are derived by differentiating the objective function J w.r.t. the membership degrees u_{ij} and w.r.t. the prototypes $\vec{\beta}_i$. Therefore they vary depending on what additional information is included in the prototypes (size and shape) and how the distances are computed. Each choice yields a different algorithm.

Of course, the simplest choice is to use only cluster centers and a Euclidean distance function (thus implicitly fixing that the clusters are spheres of equal size). The result is the well-known fuzzy C means algorithm [2]. This algorithm, however, is very inflexible and often leads to an insufficient fit of the data. In addition, it cannot easily be interpreted probabilistically. Therefore, in the following, we discuss a more flexible algorithm that is explicitly based on a probabilistic model.

The Fuzzy Maximum Likelihood Estimation (FMLE) [8] is based on the assumption that process that generated the dataset can be modeled by m n-dimensional normal distributions, where m is the number of clusters. To represent the necessary parameters, each cluster prototype is a triple $\vec{\beta}_i = (\vec{\mu}_i, \mathbf{\Sigma}_i, p_i)$, where $\vec{\mu}_i$ is the cluster center (expected value) of the cluster c_i, $\mathbf{\Sigma}_i$ is the $n \times n$ covariance matrix, which describes size and shape of the cluster, and p_i is the (prior) probability of the cluster (i.e., p_i determines the relative frequency of data points that are generated from the distribution underlying the cluster). The set of all cluster prototypes defines a complex probability density function on the n-dimensional domain under consideration, from which the probability densities at the data points in the dataset X can be determined.

The FMLE algorithm classifies the data using a maximum likelihood approach. That is, it tries to determine the parameters of the cluster prototypes in such a way that the probability of the dataset (or, to be more precise, the sum of the probability densities at the sample data points) is maximized. The rationale underlying this is that prior to observing the data all sets of prototypes are equally likely. With this assumption, the posterior probability of the dataset given the prototypes is a direct measure of the probability of the prototypes given the dataset (simply apply Bayes rule).

The distance measure used in the FMLE algorithm is inversely proportional to the probability density as defined by a cluster prototype. To be more precise, it is

$$d(\vec{x}_j, \vec{\beta}_i) = \text{const.}\cdot \left(\frac{p_i}{\sqrt{|\mathbf{\Sigma}_i|(2\pi)^n}} \exp\left(-\frac{1}{2}(\vec{x}_j - \vec{\mu}_i)^T \mathbf{\Sigma}_i^{-1}(\vec{x}_j - \vec{\mu}_i)\right)\right)^{-1}.$$

Unfortunately, if the fuzzy maximum likelihood estimation algorithm is applied exactly in the way outlined above, it tends to be unstable, mainly because of the large number of degrees of freedom. A serious problem that occurred frequently during our experiments was that one of the clusters became very small, with the shape either a sphere or a very thin and long ellipsoid. Therefore we restricted the relative size of the clusters by constraining the relative values of the determinants: If they deviate more than by a factor of three from the average, they are forced back into the range defined by the average and this factor. This lead to a much more stable behavior.

If we compare the FMLE algorithm to a naive Bayes classifier by assuming that the attributes are independent given the clusters, then the clusters are defined by their probability, their centers, and the variances for each attribute (i.e., in the covariance matrix all non-diagonal elements are zero). Intuitively, with this assumption, the clusters are axis-parallel (see above). In this case the degree of membership of a datum to a cluster is computed in much the same way as a naive Bayes classifier computes the conditional class probabilities. Thus, an axis-parallel variant of the FMLE algorithm [12] can be seen as a direct analog of a naive Bayes classifier. The only difference, of course, is that a naive Bayes classifier already knows the classes the cases belong to, whereas the clustering algorithm tries to find a good partitioning into classes. Nevertheless, if there is class information, and if the attributes convey information about the class, the class information can often be used to assess the quality of a clustering result.

If the assumption that the attributes are independent given the class does not hold, the normal version of the fuzzy maximum likelihood estimation algorithm can be applied. Since it uses full covariance matrices, dependences between the attributes can be taken into account.

As an example we take another look at the iris data. If we use the axis-parallel variant of the FMLE algorithm the clusters found are hardly distinguishable from the naive Bayes clusters shown on the left in figure 2. The result of the general (not axis-parallel) algorithm (which for reasons of space is not shown), however, differs considerably from the result obtained with a probabilistic network that takes into account the covariance of the two measures. A possible explanation for this behavior is that the fuzzy maximum likelihood estimation algorithm does not use any class information. Without such information the clusters found are much more likely than the probabilistic network clusters.

However, the fact that no class information is taken into account can also turn out to be an advantage, since we are not bound to using just one cluster per class. We may choose the number of clusters freely, and if we take a closer look at the iris data, a choice of four clusters suggests itself. Indeed, with this number of clusters the fuzzy maximum likelihood estimation algorithm yields a model that excellently fits the data as shown in figure 5. The iris virginica cases have been divided into two clusters, which is what a human would do under these circumstances. It has to be admitted though that even with the constraint

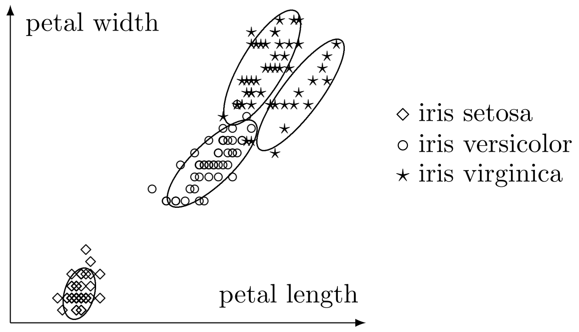

Fig. 5. Density functions generated by the fuzzy maximum likelihood estimation algorithm for the iris data, four clusters. The ellipses are the 2σ-boundaries of the density functions.

on the cluster sizes introduced above, the FMLE algorithm is not completely stable and that this is not the only classification we obtained. Fortunately, the different results can easily be ranked by simply computing the value of the objective function. Since this function has to be minimized, a smaller value indicates a better solution. The value of the objective function for the result shown in figure 5 is only half as large as the value for any other result we obtained and thus this solution can clearly be regarded as the one to be chosen.

This example indicates how naive Bayes classifiers and maybe also probabilistic networks can profit from fuzzy clustering. Using more than one cluster per class can often improve the fit to the data. In the future we plan to investigate combinations of the discussed methods.

V. Conclusions

In this paper we discussed the relationship between naive Bayes classifiers, probabilistic networks, and fuzzy cluster analysis. As we hope to have made clear, both probabilistic networks and the FMLE algorithm can be seen as generalizations of naive Bayes classifiers. However, they generalize them to different degrees. Whereas probabilistic networks only remove the requirement that the multivariate normal distributions have to be axis-parallel (by taking covariances into account), fuzzy clustering does not only this, but also lets us use more than one cluster per class. Since the normal distribution assumption, even if covariances are taken into account, is not always appropriate, this opens up a route to enhance the capabilities of the former methods. The idea is simply to split one or more classes into pseudo-subclasses, each with a multivariate normal distribution of its own. Fuzzy clustering methods may be used to find a good split into subclasses, as the example shown clearly indicates.

References

[1] S.K. Andersen, K.G. Olesen, F.V. Jensen, and F. Jensen. HUGIN — A Shell for Building Bayesian Belief Universes for Expert Systems. *Proc. 11th Int. J. Conf. on Artificial Intelligence (IJCAI'89, Detroit, MI, USA)*, 1080–1085. Morgan Kaufman, San Mateo, CA, USA 1989

[2] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, NY, USA 1981

[3] J.C. Bezdek and S.K. Pal. *Fuzzy Models for Pattern Recognition — Methods that Search for Structures in Data.* IEEE Press, Piscataway, NJ, USA 1992

[4] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and Unsupervised Discretization of Continuous Features. *Proc. 12th Int. Conf. on Machine Learning (ICML'95, Lake Tahoe, CA, USA)*, 194–202. Morgan Kaufman, San Mateo, CA, USA 1995

[5] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* J. Wiley & Sons, New York, NY, USA 1973

[6] B.S. Everitt. *Cluster Analysis.* Heinemann, London, United Kingdom 1981

[7] K.J. Ezawa and S.W. Norton. Knowledge Discovery in Telecommunication Services Data Using Bayesian Network Models. *Proc. 1st Int. Conf. on Knowledge Discovery and Data Mining (KDD'95, Montreal, Canada)*, 100–105. AAAI Press, Menlo Park, CA, USA 1995

[8] I. Gath and A.B. Geva. Unsupervised Optimal Fuzzy Clustering. *IEEE Trans. Pattern Anal. Mach. Intelligence* 11:773–781. IEEE Press, Piscataway, NJ, USA, 1989

[9] J. Gebhardt and R. Kruse. POSSINFER — A Software Tool for Possibilistic Inference. In: D. Dubois, H. Prade, and R. Yager, eds. *Fuzzy Set Methods in Information Engineering: A Guided Tour of Applications*, 407–418. J. Wiley & Sons, New York, NY, USA 1996

[10] I.J. Good. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods.* MIT Press, Cambridge, MA, USA 1965

[11] D. Heckerman. *Probabilistic Similarity Networks.* MIT Press, Cambridge, MA, USA 1991

[12] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis.* J. Wiley & Sons, Chichester, England 1999

[13] R. Kruse, E. Schwecke, and J. Heinsohn. *Uncertainty and Vagueness in Knowledge-based Systems: Numerical Methods (Series Artificial Intelligence).* Springer, Berlin, Germany 1991

[14] R. Kruse, J. Gebhardt, and F. Klawonn. *Foundations of Fuzzy Systems.* J. Wiley & Sons, Chichester, England 1994

[15] P. Langley, W. Iba, and K. Thompson. An Analysis of Bayesian Classifiers. *Proc. 10th Nat. Conf. on Artificial Intelligence (AAAI'92, San Jose, CA, USA)*, 223–228. AAAI Press and MIT Press, Menlo Park and Cambridge, CA, USA 1992

[16] P. Langley and S. Sage. Induction of Selective Bayesian Classifiers. *Proc. 10th Conf. on Uncertainty in Artificial Intelligence (UAI'94, Seattle, WA, USA)*, 399–406. Morgan Kaufman, San Mateo, CA, USA 1994

[17] S.L. Lauritzen and D.J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society, Series B*, 2(50):157–224. Blackwell, Oxford, United Kingdom 1988

[18] D. Nauck, F. Klawonn, and R. Kruse. *Foundations of Neuro-Fuzzy Systems.* J. Wiley & Sons, Chichester, England 1997

[19] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference (2nd edition).* Morgan Kaufman, San Mateo, CA, USA 1992

[20] A. Saffiotti and E. Umkehrer. PULCINELLA: A General Tool for Propagating Uncertainty in Valuation Networks. *Proc. 7th Conf. on Uncertainty in Artificial Intelligence (UAI'91, Los Angeles, CA, USA)*, 323–331. Morgan Kaufman, San Mateo, CA, USA 1991

[21] G. Shafer and P.P. Shenoy. *Local Computations in Hypertrees (Working Paper 201).* School of Business, University of Kansas, Lawrence, KS, USA 1988

[22] P.P. Shenoy. *Valuation-based Systems: A Framework for Managing Uncertainty in Expert Systems (Working Paper 226).* School of Business, University of Kansas, Lawrence, KS, USA 1991

[23] J. Whittaker. *Graphical Models in Applied Multivariate Statistics.* J. Wiley & Sons, Chichester, England 1990