Bayesian Classification*

Peter Cheeseman
RIACS

Matthew Self, Jim Kelly, Will Taylor, Don Freeman
Sterling Software

John Stutz
NASA

NASA Ames Research Center
Mail Stop 244-17
Moffett Field, CA 94035

Abstract

This paper describes a Bayesian technique for unsupervised classification of data and its computer implementation, AutoClass. Given real valued or discrete data, AutoClass determines the most probable number of classes present in the data, the most probable descriptions of those classes, and each object's probability of membership in each class. The program performs as well as or better than other automatic classification systems when run on the same data and contains no *ad hoc* similarity measures or stopping criteria. AutoClass has been applied to several databases in which it has discovered classes representing previously unsuspected phenomena.

1 Introduction

AutoClass, an automatic classification program, searches for classes in data using Bayesian statistical techniques. It defines classes not as partitions of the data but as probabilistic descriptions of processes represented in the data. From these descriptions, one can determine the probability that each object is a member of each class. The resulting classification system has several important advantages over most previous work:

- AutoClass automatically determines the most probable number of classes. The classes found represent actual structure in the data. Given random data, AutoClass discovers a single class.
- Bayes's theorem is all that is required to perform classification. No ad hoc similarity measure, stopping rule, or clustering quality criterion is needed. Decision theory applies directly to the probability distributions calculated by AutoClass.
- Classification is probabilistic. Class descriptions and assignments of objects to classes are given as probability distributions. The resulting "fuzzy" classes capture the common sense notion of class membership better than a categorical classification.
- Real valued and discrete attributes may be freely mixed, and any attribute values may be missing. "Tree valued" attributes can be easily incorporated into the AutoClass model as well.
- Classifications are invariant to changes of the scale or origin of the data.

*This work partially supported by NASA grant NCC2-428

2 Theory

When classifying a database, AutoClass does not attempt to partition the data into classes, but rather computes probabilistic descriptions of classes which account for the observed data. In order to find classes in a set of data, we make explicit declarations of how members of a class will be distributed in the data in the form of parameterized probabilistic class model functions. For instance, in classifying a database of cars, we might assume that the weights of cars in a particular class will be distributed normally with a mean of 3000 pounds and a standard deviation of 100 pounds. Our class model function in this case is a Gaussian curve. Once the classes are specified in this way, we can find the probability of the data having come from such a set of classes by simple probability formulas. Finding the best classification is then a matter of varying the class parameters—for instance, adjusting the mean and standard deviation—until they are maximally predictive of the data. Classification has long been studied in these terms as the theory of finite mixtures. Everitt and Hand [1981] provide an excellent review containing over 200 references.

AutoClass is an implementation of the Bayesian solution to the finite mixture problem. We begin with an uninformative prior probability distribution over the classification parameters (which expresses our *a priori* ignorance of the parameters) and then update this distribution by using the information in the database to calculate the posterior probability distribution of the parameters. This posterior distribution allows us to determine both the most probable classification parameters for a given number of classes as well as the most probable number of classes present in the data. From this information it is also possible to calculate the probability that each object is a member of each class. Note that it is possible to determine the parameters of strongly overlapping classes accurately, although very few of the objects can be assigned to any class with high probability.

In addition to providing the database, the user selects an appropriate class model. For real valued variables, for example, the default model is a Gaussian distribution. AutoClass then calculates the optimal values of the parameters for a given number of classes and the probability that each number of classes is actually present in the data. As final output, AutoClass provides the most probable number of classes, the most probable values of the classification parameters for that number of classes, and also the probability of membership of each object in each class.

In order to make any headway into classification, and indeed to give meaning to the term, one must define what one means by a class. We do so mathematically through the class model functions. By committing ourselves to spe-

cific functions, we are not assuming the functions describe the actual classes any more than the act of looking for classes assumes that classes exist. Rather, we are setting forth precisely the question we wish to ask: "What classes of the given form can be found in the data?"

The current AutoClass program (AutoClass II) looks for classes in which attributes vary independently within a class. It models real-valued attributes with Gaussian probability distributions and discrete attributes with lists of outcome probabilities. We phrased our classification question in these terms to simplify implementation, with the realization that ignoring attribute dependence neglects potentially useful information. Working within this framework, we have found meaningful structure in many databases, as Section 4 attests.

AutoClass uses a Bayesian variant of Dempster and Laird's EM Algorithm [Dempster *et al.*, 1977] to search for the maximum of the posterior distribution of the classification parameters and approximates the distribution about this maximum. AutoClass also includes heuristic techniques for avoiding local maxima in the search. Although local maxima present a difficult problem in practice, they are an algorithmic concern and require no additional theory. Details of the Bayesian theory of finite mixtures appear in the Appendix. The AutoClass algorithm is described thoroughly by Cheeseman *et al.* [1988]

3 Discussion

It is important to point out that we do *not* assume that the classification parameters or the number of classes are "random variables." They have definite but unknown values which we must infer. The prior distributions used do not represent a frequency distribution of the parameters, but rather the state of knowledge of the observer (in this case AutoClass) before the data are observed. Thus there can be no "true values of the prior probabilities" as Duda and Hart suggest [1973], since prior probabilities are a function of the observer, not of the world. Although Cox gave the first full explanation of this issue in 1946 [Cox, 1946], it remains a source of confusion today.[1]

Bayesian methods have often been rejected due to their use of prior distributions, because of the belief that priors taint the analysis with personal biases. It is possible to use priors that are uninformative and completely impersonal.[2] These are invariant to any change of scale or origin, so in no way do they express any *a priori* opinions or biases. Rather, they express complete *a priori* ignorance of the parameters (as defined by specific invariance criteria).

On the other hand, the ability to incorporate prior knowledge can be of great use when such information is available. Informative priors are often mathematically simpler than their uninformative brethren, and for this reason AutoClass uses a weak, informative prior which introduces little bias. AutoClass could be easily extended to include strong prior knowledge, if it is available, whereas many non-Bayesian approaches would have difficulty incorporating such knowledge smoothly.

AutoClass can be used to learn from examples. If the program is given a set of objects pre-classified by a teacher, it can form descriptions of the specified classes and use these to classify new objects. Furthermore, it can estimate missing parameter values from its classification based on the values present. Thus supervised learning can be combined with unsupervised learning in the same system, using the same theory.

Development of AutoClass III is underway. It will include exponential distributions for real attributes and multivariate distributions that will make use of dependence between attributes. We are also developing the theory for automatic selection of class distributions, allowing the system to take advantage of increased model complexity when the data justify estimation of the additional parameters. Thus, simple theories (with correspondingly few parameters) can give way to more complex theories as the amount of data increases. The theory for such model selection is very similar to the selection of the number of classes.

4 Results

AutoClass has classified data supplied by researchers active in various domains and has yielded some new and intriguing results:

- **Iris Database**

Fisher's data on three species of iris [Fisher, 1936] are a classic test for classification systems. AutoClass discovers the three classes present in the data with very high confidence, although not all of the cases can be assigned to their classes with certainty. Wolfe's NORMIX and NORMAP [Wolfe, 1970] both incorrectly found four classes, and Dubes's MH index [Dubes, 1987] offers only weak evidence for three clusters.

- **Soybean Disease Database**

AutoClass found the four known classes in Stepp's soybean disease data, providing a comparison with Michalski's CLUSTER/2 system [Michalski and Stepp, 1983a]. AutoClass's class assignments exactly matched Michalski's—each object belonged overwhelmingly to one class, indicating exceptionally well separated classes for so small a database (47 cases, 35 attributes).

- **Horse Colic Database**

AutoClass analyzed the results of 50 veterinary tests on 259 horses and extracted classes which provided reliable disease diagnoses, although almost 40% of the data were missing.

- **Infrared Astronomy Database**

The Infrared Astronomical Satellite tabulation of stellar spectra is not only the largest database AutoClass has assayed (5,425 cases, 94 attributes) but the least thoroughly understood by domain experts. AutoClass's results differed significantly from previous analyses. Preliminary evaluations of the new classes by infrared astronomers indicate that the hitherto unknown classes have important physical meaning. The AutoClass infrared source classification is the basis of a new star catalog to appear shortly.

[1] See Jaynes [1986] for a recent discussion of the nature of Bayesian inference and its relationship to other methods of statistical inference.

[2] See Jaynes [1968] for a lucid description of uninformative priors.

We are actively collecting and analyzing other databases which seem appropriate for classification, including an AIDS database and a second infrared spectral database.

5 Comparison with Other Methods

Several different communities are interested in automatic classification, and we compare AutoClass to some existing methods:

• Maximum Likelihood Mixture Separation

AutoClass is very similar to the maximum likelihood methods used to separate finite mixtures as described in the statistical pattern recognition literature. The mathematical statement of the problem is identical to that discussed by Duda and Hart [1973] and by Everitt and Hand [1981]. The primary difference lies in AutoClass's Bayesian formulation, which removes singularities from the search space and provides a more effective method for determining the number of classes than existing methods based on hypothesis testing. A more detailed comparison of AutoClass to maximum likelihood methods is given by Cheeseman *et al.* [1988]

• Cluster Analysis

Cluster analysis and AutoClass's finite mixture separation differ fundamentally in their goals. Cluster analysis seeks classes which are groupings of the data points, definitively assigning points to classes; AutoClass seeks descriptions of classes that are present in the data, and never assigns points to classes with certainty.

The other major difference lies in the definition of a class. The AutoClass method defines a class explicitly with model functions and then derives the optimal class separation criterion using Bayes's theorem. Cluster analysis techniques define a class indirectly by specifying a criterion for evaluating clustering hypotheses, such as maximizing some form of intra-class similarity.

• Conceptual Clustering

Both AutoClass and conceptual clustering methods seek descriptions of the clusters rather than a simple partitioning of the objects. The main difference between the methods is the choice of concept language: AutoClass uses a probabilistic description of the classes, while Michalski and Stepp [1983b] use a logical description language. The logic-based approach is particularly well suited to logically "clean" applications, whereas AutoClass is effective even when the data are noisy or the classes overlap substantially.

Conceptual clustering techniques specify their class definitions with a "clustering quality criterion" such as "category utility." [Fisher, 1987] As with cluster analysis, these criteria impose constraints on what clusterings are desired rather than on the nature of the actual clusters. This may reflect a difference in goals since Langley's CLASSIT [Langley *et al.*, 1987] and Michalski's CLUSTER/2 [Michalski and Stepp, 1983a] programs seek explicitly to emulate human classification, which is a more difficult problem than AutoClass addresses.

• Minimum Message Length Method

A classification method based on minimum total message length (MML) was introduced 20 years ago [Wallace and Boulton, 1968] and has been considerably extended since then. [Wallace and Freeman, 1987] This method searches for the classification that can be encoded in the fewest bits, where the encoded message consists of two parts: the information required to describe the class parameters (i.e., the particular classification model) and the information required to encode the data given the parameters. Because this method tries to minimize the *total* message length, there is a built-in tradeoff between the complexity of the model (the information required to describe the classes) and the fit to the data (the information required to encode the data given the classes). This is the same tradeoff given by the Bayesian approach, and in fact the minimum message length criterion is a very good approximation to the Bayesian criterion. See Georgeff [Georgeff and Wallace, 1984] for details. Note that the MML method requires the parameters to be estimated to an optimal accuracy that depends on the data.

6 Conclusion

We have developed a practical and theoretically sound method for determining the number of classes present in a mixture, based solely on Bayes's theorem. Its rigorous mathematical foundation permits the assumptions and definitions involved to be stated clearly and analyzed carefully. The AutoClass method determines the number of classes better than existing mixture separation methods do and also compares favorably with cluster analysis and conceptual clustering methods.

Appendix

This appendix presents the Bayesian theory of finite mixtures, the mathematical basis of the AutoClass algorithm.

In the theory of finite mixtures, each datum is assumed to be drawn from one of m mutually exclusive and exhaustive classes. Each class is described by a *class distribution*, $p(x_i \mid x_i \in C_j, \vec{\theta}_j)$, which gives the probability distribution of the attributes of a datum if it were known to belong to the class C_j. These class distributions are assumed to be parameterized by a *class parameter vector*, $\vec{\theta}_j$, which for a normal distribution would consist of the class mean, μ_j, and variance, σ_j^2. The probability of an object being drawn from class j is called the *class probability* or mixing proportion, π_j. Thus, the probability distribution of a datum drawn from a mixture distribution is

$$p(x_i \mid \vec{\theta}, \vec{\pi}, m) = \sum_{j=1}^{m} \pi_j \, p(x_i \mid x_i \in C_j, \vec{\theta}_j). \qquad (1)$$

We assume that the data are drawn from an exchangeable (static) process—that is, the data are unordered and are assumed to be independent given the model. Thus, the *joint* probability distribution of a set of n data drawn from a finite mixture is

$$p(\vec{x} \mid \vec{\theta}, \vec{\pi}, m) = \prod_{i=1}^{n} p(x_i \mid \vec{\theta}, \vec{\pi}, m). \qquad (2)$$

For a given value of the class parameters, we can calculate the probability that an object belongs to each class using Bayes's theorem,

$$p(x_i \in C_j \mid x_i, \vec{\theta}, \vec{\pi}, m) = \frac{\pi_j \, p(x_i \mid x_i \in C_j, \vec{\theta}_j)}{p(x_i \mid \vec{\theta}, \vec{\pi}, m)}. \quad (3)$$

Thus, the classes are "fuzzy" in the sense that even with perfect knowledge of an object's attributes, it will only be possible to determine the probability that it is a member of a given class.

We break the problem of identifying a finite mixture into two parts: determining the classification parameters for a given number of classes, and determining the number of classes. Rather than seeking an *estimator* of the classification parameters (the class parameter vectors, $\vec{\theta}$, and the class probabilities, $\vec{\pi}$), we seek their full posterior probability distribution. The posterior distribution is proportional to the product of the prior distribution of the parameters, $p(\vec{\theta}, \vec{\pi} \mid m)$, and the likelihood function, $p(\vec{x} \mid \vec{\theta}, \vec{\pi}, m)$:

$$p(\vec{\theta}, \vec{\pi} \mid \vec{x}, m) = \frac{p(\vec{\theta}, \vec{\pi} \mid m) \, p(\vec{x} \mid \vec{\theta}, \vec{\pi}, m)}{p(\vec{x} \mid m)}, \quad (4)$$

where $p(\vec{x} \mid m)$ is simply the normalizing constant of the posterior distribution, and is given by

$$p(\vec{x} \mid m) = \int\int p(\vec{\theta}, \vec{\pi} \mid m) \, p(\vec{x} \mid \vec{\theta}, \vec{\pi}, m) \, d\vec{\theta} \, d\vec{\pi}. \quad (5)$$

To solve the second half of the classification problem (determining the number of classes) we calculate the posterior distribution of the number of classes, m. This is proportional to the product of the prior distribution, $p(m)$, and the pseudo-likelihood function, $p(\vec{x} \mid m)$,

$$p(m \mid \vec{x}) = \frac{p(m) \, p(\vec{x} \mid m)}{p(\vec{x})}. \quad (6)$$

The pseudo-likelihood function is just the normalizing constant of the posterior distribution of the classification parameters (Equation 5). Thus, to determine the number of classes, we first determine the posterior distribution of the classification parameters for each possible number of classes. We then marginalize (integrate) out the classification parameters from the estimation of the number of classes—in effect, treating them as "nuisance" parameters.

In general, the marginalization cannot be performed in closed form, so AutoClass searches for the maximum of the posterior distribution of the classification parameters (using a Bayesian variant of Dempster and Laird's EM Algorithm [Dempster *et al.*, 1977]) and forms an approximation to the distribution about this maximum. Including the search, the algorithm is roughly linear in the amount of data multiplied by the number of classes. See Cheeseman *et al.* [1988] for full details of the AutoClass algorithm.

Note that in finding the posterior probability distribution over the number of classes, we are comparing models with different numbers of parameters. Maximum likelihood methods always favor models with more parameters, because these extra parameters can be adjusted to fit the data better. Bayesian model comparison, on the other hand, automatically penalizes additional parameters unless they substantially improve the fit to the data. That is, Bayesian model comparison has a built-in tradeoff between complexity of the model and the fit to the data. In the classification model, Equations 5 and 6 give this tradeoff. In particular the probability in Equation 6 does not automatically grow with additional classes, because the additional classes introduce additional parameters and so increase the dimensionality of the integral in the denominator (Equation 5). Unless the likelihood inside the integral is strongly increased by these additional parameters, the increased dimensionality will lower the total probability.

References

[Cheeseman *et al.*, 1988] Peter Cheeseman, Don Freeman, James Kelly, Matthew Self, John Stutz, and Will Taylor. Autoclass: a Bayesian classification system. In *Proceedings of the Fifth International Conference on Machine Learning*, 1988.

[Cox, 1946] R. T. Cox. Probability, frequency, and reasonable expectation. *American Journal of Physics*, 17:1–13, 1946.

[Dempster *et al.*, 1977] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[Dubes, 1987] Richard C. Dubes. How many clusters are best? — an experiment. *Pattern Recognition*, 20(6):645–663, 1987.

[Duda and Hart, 1973] Richard O. Duda and Peter E. Hart. *Pattern Recognition and Scene Analysis*, chapter 6. Wiley-Interscience, 1973.

[Everitt and Hand, 1981] B. S. Everitt and D. J. Hand. *Finite Mixture Distributions. Monographs on Applied Probability and Statistics*, Chapman and Hall, London, England, 1981. Extensive Bibliography.

[Fisher, 1987] D. H. Fisher. Conceptual clustering, learning from examples, and inference. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 38–49, Morgan Kaufmann, 1987.

[Fisher, 1936] R. A. Fisher. Multiple measurments in taxonomic problems. *Annals of Eugenics*, VII:179–188, 1936.

[Georgeff and Wallace, 1984] M. P. Georgeff and C. S. Wallace. A general selection criterion for inductive inference. In T. O'Shea, editor, *ECAI-84: Advances in Artificial Intelligence*, pages 473–482, Elsevier, Amsterdam, 1984.

[Jaynes, 1968] Edwin T. Jaynes. Prior probabilities. *IEEE Transactions on Systems and Cybernetics*, SSC-4(3):227–241, September 1968. (Reprinted in [Jaynes, 1983]).

[Jaynes, 1983] Edwin T. Jaynes. *Papers on Probability, Statistics and Statistical Physics.* Volume 158 of *Synthese Library*, D. Reidel, Boston, 1983.

[Jaynes, 1986] Edwin T. Jaynes. Bayesian methods: general background. In James H. Justice, editor, *Maximum Entropy and Bayesian Methods in Applied Statistics*, pages 1–25, Cambridge University Press, Cambridge, Massachusetts, 1986.

[Langley *et al.*, 1987] Pat Langley, John H. Gennari, and Wayne Iba. Hill-climbing theories of learning. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 312–323, Morgan Kaufmann, 1987.

[Michalski and Stepp, 1983a] Ryszard S. Michalski and Robert. E. Stepp. Automated construction of classifications: conceptual clustering versus numerical taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:396–410, 1983.

[Michalski and Stepp, 1983b] Ryszard S. Michalski and Robert E. Stepp. Learning from observation: conceptual clustering. In Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, chapter 11, Tioga Press, Palo Alto, 1983.

[Wallace and Boulton, 1968] C. S. Wallace and D. M. Boulton. An information measure for classification. *Computer Journal*, 1:185–195, 1968.

[Wallace and Freeman, 1987] C. S. Wallace and P. R. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society, Series B*, 49(3):223–265, 1987.

[Wolfe, 1970] John H. Wolfe. Pattern clustering by multivariate mixture analysis. *Multivariate Behavioural Research*, 5:329–350, July 1970.