Pattern Classification: A Survey and Comparison

Ayşe Küçükyılmaz
Department of Computer Engineering,
Bilkent University, 06800, Ankara, Turkey
E-mail: aysek@cs.bilkent.edu.tr

April 7, 2005

Abstract

This paper presents an overview of several pattern classification methods in literature. The overview is accompanied by comparisons of these algorithms on different datasets. Deciding on the method for classification is often a difficult task because the quality of the results vary considerably with the input data. The aim of this work is to experiment several methods with different data to find out the characteristics of the methods.

1 Introduction

With the developments in technology, automation has gained an important place in our lives. As the mankind has seen the things computers can do, and how much they facilitate our lives, the capabilities of computers are begun to be questioned. Could the computers see, or could they understand? And maybe the most important of all questions in computer literature, could they learn?

There has been several techniques suggested and used for machine learning. Although it has been long debated whether these are really learning or not in the philosophical sense, the computer scientists agreed to use this terminology. In this paper, I will talk about some of these techniques within the scope of pattern classification.

Pattern recognition is the problem of identifying smaller, meaningful patterns in a bigger pattern. Pattern classification is interested in labeling these smaller patterns, and is an important part of pattern recognition. *Statistical* pattern recognition systems [3] possess the structure given in Figure 1.

This figure illustrates the "modes" of he recognition system: training and testing. The preprocessing unit gets the input data, segments it into meaningful parts, and eliminates noise. When training, the feature extracting unit gathers suitable features, which will help the classifier to partition the feature space the best way possible. The learning module is fed with the output of this unit

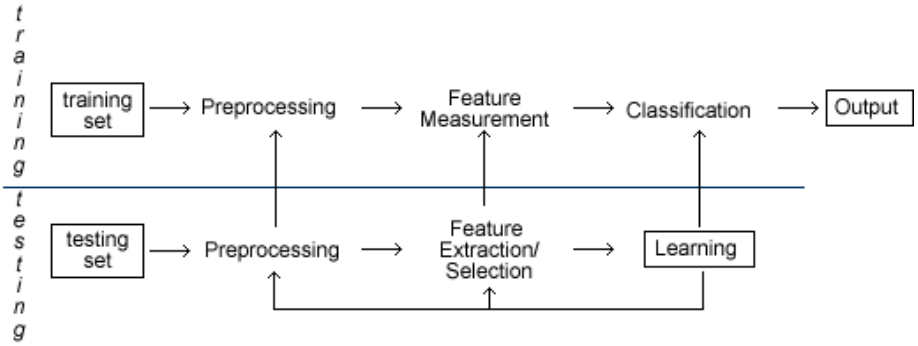

Figure 1: The components of a statistical pattern recognition system.

in order to train the classifier unit. The training may take several iterations to optimize the selected features. When testing, upon receiving the measured features, the classifier identifies the pattern classes.

The classification component here is of special interest, because its success, i.e. the ability to correctly classify the test instances, depend mostly on the output of the feature extraction algorithm. Currently, there are no feature extraction algorithms that perform good globally. That is, the quality of the extracted features depend on the input data. Classification aims to group objects in a category, based on the variance in their feature values. Objects in the same category inhibit similar feature values; while for the ones in distinct categories, the similarity decreases. The classifiers are designed to handle several discrepancies in feature sets, yet each have its pros and cons when dealing with specific feature sets.

The organization is as follows: In section 2, I give information on several classifiers. The classifiers are categorized according to the design methodology; as using similarity maximization, probability, and geometric information on deciding. Here, I will talk about the first two categories along with the notion of decision trees. Later in section 3 the experiments done using example classifiers for each category on different datasets will be given, and finally in section 4, I will go over the results to conclude.

2 Pattern Classification

There are mainly three classes of classifiers [3]. Each has its advantages, though the outcoming result depends mostly on the training set, and the feature selection algorithm. These classes consist of classifiers that depend on similarity maximization methods, probabilistic methods, and geometric methods, respectively.

The first class of classifiers have some similarity metrics and assign class

labels for maximizing the similarity.

Probabilistic methods, for which the Bayesian classifier is the most known, depend on the prior probabilities of classes and class-conditional densities of the instances. In addition to Bayesian classifiers, logistic classifiers belong to this type of classifiers. The logistic classifiers deal with unknown parameters based on the maximum-likelihood (see [8] approach. Further details on logistic classifiers can be found in [9].

This survey doesn't cover geometric classifiers, which build decision boundaries by directly minimizing the error criterion, since no related experiments are supplied. An example to these classifiers is Fisher's linear discriminant, which mainly aim to reduce the size of the feature space to lower dimensions in case of a huge number of features. It minimizes the mean squared error between the class labels and the tested instance. Additionally, neural networks are examples of geometric classifiers.

2.1 Similarity Maximization Methods

The first type of classifiers that we will talk about uses the similarity between patterns to decide on a good classification. The question is how to define similarity. The nearest mean classifiers define the features of a class as a vector and represent the class with the mean of the elements of this vector. Thus, any unlabeled vector of features will be classified as the class with nearest mean value.

Template matching uses a template for defining class labels, and tries to find the most similar template for classification. However, there are problems with this approach, for instance, if we want to classify face images, we need to supply a template for each face label. Also, scaling affect the matching, with improper data, the algorithm might fail to produce good results.

Another important classifier of this type uses the Nearest Neighbor (NN) Algorithm [5, 6]. The data is represented as points in space, and classification is done based on the, i.e. Euclidean, distance of the data to the labeled classes. For the k-NN, the classifier checks the k nearest points and decides in favor of the majority.

2.2 Probabilistic Methods

The most well known of probabilistic methods makes use of Bayesian Decision Theory. The decision rule assign class labels to that having the *maximum posterior probability*. The posterior can be calculated by the well-known Bayes rule:

$$posterior = \frac{likelihood \times prior}{evidence}. \tag{1}$$

Let's define the variables. $w = w_i$ is the *state of the nature*, i.e instance belongs to class i. Hence $P(w_i)$ is the prior probability that the instance belongs to class i. $p(x|w_i)$ is the class-conditional probability density function: the

density for x given that the instance is of class i. Finally $p(x)$ is defined as $\sum p(x|W_j) \times P(w_j)$ over all classes.

Given the definitions equation 1 is equivalent to

$$P(w_j|x) = \frac{p(x|w_j) \times P(w_j)}{p(x)}. \tag{2}$$

The classification is done in favor of the j^{th} class, if $P(w_j|x) > P(w_i|x)$ $\forall w_i \in C$, $w_i \neq w_j$, where C is the set of classes ($w_j \in C$).

Naive Bayes, a.k.a Idiot Bayes, uses Bayesian Decision Theory, while presuming a conditional independence, rather "naively". For instance, with the naive model, we can calculate the full joint distribution of causes and effects as

$$P(c, e_1, e_2, ..., e_n) = P(c) \prod_i P(e_i|c), \tag{3}$$

where c is the cause, and e_i are the conditionally independent effects ($i = 1..n$).

Finally, Bayesian Belief Nets [12, 13, 14] represent the functional dependencies and independencies among model variables, i.e. features. Whenever some parameters take some values, the nodes of the network are affected and take a probability value, by the Bayes' rule.

2.3 Decision Trees

The decision trees, take the instance described by its features as input, and outputs a *decision*, denoting the class information in our case. Each node denotes a feature, and each iteration we go down to the lower depth, selecting a child node depending on the feature value for the particular instance. There are several issues of decision trees, such as how to create a good one. There are several decision tree classifiers such as ID3 or C4.5, which will be demonstrated in section 3.

3 Experiments

The experiments explained in this section are realized using the WEKA software v.3.4 [10], with datasets taken from the UCI Repository of machine learning databases [11]. The training and test sets are developed by splitting the whole dataset into two equal sized parts for all experiments.

3.1 Experiments on Iris Dataset

The iris dataset is a well-known set in the pattern recognition domain. It is exceedingly simple, with 3 classes of 50 instances each. There are four features for each instance, and no missing instances.

The dataset is tested with four classifiers: 1-NN, Naive Bayes, BayesNet and C4.5.

Table 1 gives the accuracy on each classifier.

Method	1-NN	Naive Bayes	BayesNet	C4.5
Accuracy %	96	96	94.6667	94.6667
Mean abs.err.	0.0425	0.0336	0.037	0.519
Root of mean sq.err.	0.1611	0.1606	0.1682	0.192

Table 1: The results of experiments on iris dataset.

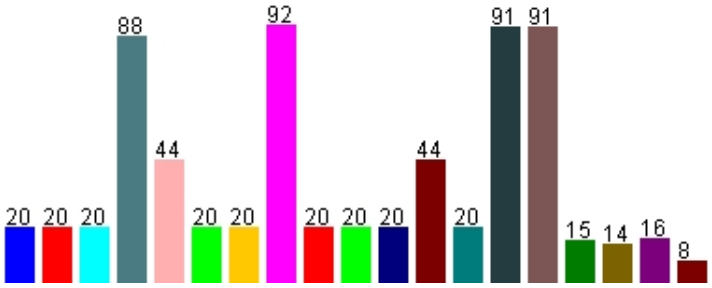

Figure 2: The number of instances per class in the soybean dataset.

3.2 Experiments on Soybean Dataset

The soybean dataset consists of 683 instances with 36 features. There are some missing features, though they are sparse. The distribution of the instances to classes is given in Figure 2.

The results are given in Table 2

3.3 Experiments on Anneal Dataset with Missing Features

Next we test the algorithms on anneal dataset consisting of 898 instances with 39 features. There are 6 classes, on which the distribution of instances is shown in Figure 3. This dataset contains plenty of missing features.

Table 3 demonstrates the results for this dataset.

Method	1-NN	Naive Bayes	BayesNet	C4.5
Accuracy %	88.0117	90.0585	90.35096667	85.6725
Mean abs.err.	0.0176	0.0124	0.0109	0.019
Root of mean sq.err.	0.1046	0.0937	0.0918	0.109

Table 2: The results of experiments on soybean dataset.

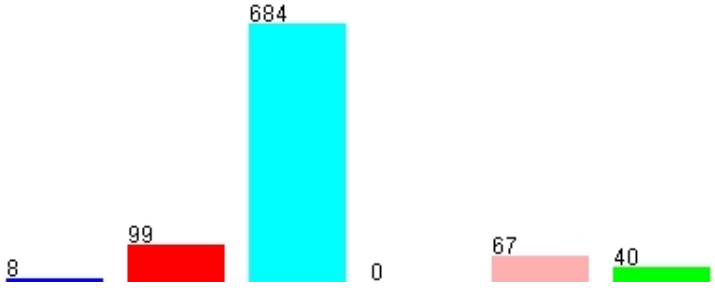

Figure 3: The number of instances per class in the anneal dataset.

Method	1-NN	Naive Bayes	BayesNet	C4.5
Accuracy %	75.5457	73.7194	89.0869	88.8641
Mean abs.err.	0.0182	0.093	0.0466	0.683
Root of mean sq.err.	0.1212	0.2543	0.1635	0.1804

Table 3: The results of experiments on anneal dataset with missing features.

3.4 Experiments on Anneal Dataset without Missing Features

In this version of the anneal dataset, the missing features are eliminated. Again, there are 6 classes and 39 features for each of the 898 instances. The distribution of instances to classes is the same as for the case with missing features.

Table 4 shows the results:

4 Conclusion

In the preceding sections, an overview of classification systems has been given. In addition, a set of classifiers, 1-NN, Naive Bayes, BayesNet and C4.5, are

Method	1-NN	Naive Bayes	BayesNet	C4.5
Accuracy %	98.441	87.0824	93.5412	98.8864
Mean abs.err.	0.0088	0.0508	0.2025	0.0054
Root of mean sq.err.	0.0718	0.2031	0.1165	0.0599

Table 4: The results of experiments on anneal dataset without the missing features.

tested with different datasets. The datasets used here are taken from the UCI Repository of machine learning databases, and have different characteristics.

The first dataset, iris, is a very simplistic one with a few classes and instances. Also, the classes are easily separable from each other geometrically. When we examine the confusion matrices, which illustrate the labeling density after classification, it is easily seen that some classes are very easy to classify. Almost all four of the classifiers perform well on this dataset, however, 1-NN and Naive Bayes perform a little better. When considering the error rates, the Naive Bayes is the best of all three. This might stem from the fact that, one of the classes are linearly separable from the other two, making the naive presumption of this algorithm a good one. Also, as noted above, the classes display great similarity within themselves. Hence, the simple becomes better.

The second dataset is the soybean set. Which is a rather large one with 683 instances, each having 36 features. The dataset covers 19 classes. The distribution of the instances to classes are not rather unbalanced, hence it is not easy to see the effect of features on each class uniformly. Naive Bayes and BayesNet perform the best on this set. The tree and the nearest neighbor algorithm suffers from the non-uniformity in the set.

The last two datasets are included to investigate the effect of missing features. The datasets are actually the same: the anneal dataset. However, in the first version, there are plenty of missing feature values, while in the second, these missing features are filled. As it can be predicted easily, the results of the first set are far worse than those of the second one, because of the uncertainty. The datasets consist of 898 instances with 39 features and 6 classes.

The anneal data with missing values perform better with BayesNet and C4.5. Since the percentage of missing features in this dataset is high, the similarity measures become obscure. When these feature values are filled 1-NN performs very good. However, Naive Bayesm although getting better, cannot classify as successfully as the others do, hence the bad performance stems from the characteristics of the dataset.

As a result, these comparisons show that no classifier can be selected *the best* globally. As explained in the previous sections, the performance of a classifier depends on many factors such as the dataset characteristics and the given features.

As an improvement to the experiments done here, different classifiers can be tested with more datasets. Hence, a notion on how the classifiers work on what conditions can be captured more coherently.

References

[1] R. O. Duda, D. G. Stork, and P. E. Hart. Pattern Classification and Scene Analysis. John Wiley & Sons, Inc., 2nd edition, 2000.

[2] S. Russell and P. Norvig. Artificial Intelligence - A Modern Approach. Prentice Hall, second edition, 2003

[3] A. K. Jain, R. P. W. Duin, and J. Mao, Statistical Pattern Recognition: A Review, IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(1):4-37, January 2000.

[4] S. R. Kulkarni, G. Lugosi, and S. S. Venkatesh, Learning Pattern ClassificationA Survey, IEEE Transactions Information Theory, vol. 44, no.6, O1998

[5] T. M. Cover and P. E. Hart, Nearest neighbor pattern classification, IEEE Transactions Information Theory, vol. IT-13, pp. 2127, 1967.

[6] Y. Chenyz Y. Hungyz C. Fuhz, Fast Algorithm for Nearest Neighbor Search Based on a Lower Bound Tree, to appear in Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada, July 2001.

[7] W. L. Buntine, Operations for Learning with Graphical Models, Journal of Artificial Intelligence Research, 2:159-225, 1994.

[8] J. A. Bilmes, A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, Technical Report TR-97-021, International Computer Science Institute, University of California, Berkeley, April 1998.

[9] J. A. Anderson, Logistic Discrimination, Handbook of Statistics. P. R. Krishnaiah and L. N. Kanal, eds., vol. 2, pp. 169-191, Amsterdam: North Holland, 1982.

[10] E. Frank, M. Hall, and L. Trigg, Weka 3 - Data Mining with Open Source Machine Learning Software in Java. http://www.cs.waikato.ac.nz/ml/weka/index.html.

[11] C. L. Blake and C. J. Merz, UCI Repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.

[12] K. P. Murphy, A Brief Introduction to Graphical Models and Bayesian Networks, May 2001.

[13] E. Charniak, Bayesian Networks Without Tears, AI Magazine, 12(4):50-63, 1991.

[14] D. Heckerman, A Tutorial on Learning With Bayesian Networks, Technical Report MSR-TR-95-06, Microsoft Research, March 1995.