COIN CLASSIFICATION USING A NOVEL TECHNIQUE FOR LEARNING CHARACTERISTIC DECISION TREES BY CONTROLLING THE DEGREE OF GENERALIZATION

Paul Davidsson
Department of Computer Science, Lund University
Lund, Sweden, S–221 00
Email: Paul.Davidsson@dna.lth.se

ABSTRACT

A novel method for learning characteristic decision trees is applied to the problem of learning the decision mechanism of coin-sorting machines. Decision trees constructed by ID3-like algorithms are unable to detect instances of categories not present in the set of training examples. Instead of being rejected, such instances are assigned to one of the classes actually present in the training set. To solve this problem the algorithm must learn characteristic, rather than discriminative, category descriptions. In addition, the ability to control the degree of generalization is identified as an essential property of such algorithms. A novel method using the information about the statistical distribution of the feature values that can be extracted from the training examples is developed to meet these requirements. The central idea is to augment each leaf of the decision tree with a subtree that imposes further restrictions on the values of each feature in that leaf.

1 INTRODUCTION

One often ignored problem for a learning system is how to know when it encounters an instance of an unknown category. In many practical applications it cannot be assumed that every category is represented in the set of training examples (i.e., they are open domains [Hut94]) and sometimes the cost of a misclassification is too high. What is needed in such situations is the ability to reject instances of categories that the system has not been trained on.

In this article we will concentrate on an application concerning a coin-sorting machine of the kind often used in bank offices.[1] Its task is to accept and sort (and count) a limited number of different types of coins (for instance, a particular country's), and to reject all other coins. The vital part of the machine is a sophisticated sensor that the coins pass one by one. The sensor measures electronically five properties (diameter, thickness, permeability, and two kinds of conductivity) of each coin, which all are given a numerical value. Based on these measurements the machine decides of which type the current coin is: if it is of a known type of coin, it is sorted, otherwise it is regarded as an unknown type and is rejected.

The present procedure for constructing the decision mechanism is carried out mostly by hand. A number of coins of each type are passed through a sensor and the measurements are recorded. The measurements are then analyzed manually by an engineer, who chose a minimum and a maximum limit for each property of each type of coin. Finally, these limits are loaded into the memory of the machine. When the machine is about to sort a new coin, it uses the limits in the following way: if the measurement is higher than the minimum limit and lower than the maximum limit for all properties of some type of coin, the coin is classified as a coin of this type. If this is not true for any known type of coin, the coin is rejected.

Thus, in the present method, which is both complicated and time consuming, it is the skill of the engineer that decides the classification performance of the coin-sorting machine. Moreover, this procedure must be carried out for every new set of machines (e.g., for each country's). In addition, there are updating problems when a new kind of coin is introduced. This is not only applicable when a new denomination is introduced, or when the appearance of an old denomination is changed. It is, for instance, not unusual that the composition of the alloy is changed. In fact, this happens often undeliberately as it is difficult to get exactly the same composition every time and, moreover, there are sometimes trace elements of other metals in the cauldron. Another kind of problems comes from the fact that it is difficult to make all the sensors exactly alike. As a consequence of this and the fact that all machines used for the same set of coins use the same limits, each sensor must be calibrated. Moreover, this calibration is not always sufficient and ser-

[1] The work presented in this paper has in part been carried out in collaboration with Scan Coin AB (Malmö, Sweden), a manufacturer of such machines.

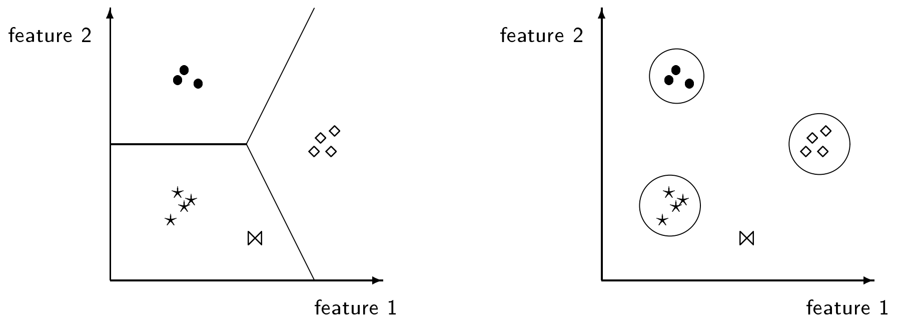

Figure 1: Discriminative (left) versus characteristic (right) category descriptions.

vice agents must sometimes be sent out to adjust the limits on particular machines.

However, if we construct a method which automatically learns the decision mechanism, it would be possible to bring down the effects of these problems. In short, the task to be solved can be described as follows: develop a system that computes the decision mechanism of a coin-sorting machine automatically from a number of examples of coins.

2 CHARACTERISTICS OF THE APPLICATION

To begin with, this is a case of learning from examples in that we can safely assume that there is a number of coins with known identities which can be used to train the system. Moreover, as the working of the machine has been described above, we can clearly divide it into two separate phases, the initial training phase and the classification phase. Thus, there is no need to make demands on incrementality upon the learning algorithm.

There are also some demands on performance. The method must be time-efficient both in the learning phase, since the long-term goal is to let the end users themselves adapt the machine to a set of coins of their choice and we cannot expect the bank clerks to be willing to spend much time training the machine, and in the classification phase, since the machine must be able sort approximately 800 coins per minute. Moreover, as the current hardware is rather limited the method also must be memory-efficient. However, as the hardware probably will be upgraded in the near future, we will not bother much about the space problem. The demands on classification performance that must be met by the learning system before it can be used in a real world application are rather tough: not more than 5% of known types of coins are allowed to be rejected and very few misclassifications (less than 0.5%) of any type of coins are accepted. Consequently, more than 99.5% of unknown types of coins should be rejected.

Another aspect of our learning task concerns a problem that often has been ignored in the machine learning research. Namely, how the learning system should "know" when it encounters an instance of an unknown category. It is for practical reasons impossible to train the learning system on every possible kind of coin (genuine or faked). Thus, we must assume that the system can be trained only on the types of coins it is supposed to accept. As been pointed out by Smyth and Mellstrom [SM92], the only way of solving this problem is to learn *generative*, or *characteristic*, category descriptions that try to capture the similarities between the members of the category. This, in contrast to learning *discriminative* descriptions that can be seen as representations of the boundaries between categories. The difference between these kinds of descriptions is illustrated in Figure 1. It shows some instances of three known categories ($\star$, $\bullet$, and $\diamond$), and examples of possible category boundaries of the concepts learned by a system using discriminative descriptions (to the left) and by a system using characteristic descriptions (to the right). In this case, a member of an unknown category ($\bowtie$) will be categorized wrongly by a system using discriminative descriptions, whereas it will be regarded as a member of a novel category by a system using characteristic descriptions.

3 RESULTS FROM A PRELIMINARY STUDY

A preliminary study was carried out by a Master's student (Mårtensson [Mar94]). Three methods were evaluated: induction of decision trees using the ID3 algorithm [Qui86], learning neural networks by the backpropagation algorithm [RHW86], and computing Bayesian classifiers [TG74]. However, since all these methods in their original versions learn discriminative descriptions, they must be modified in order to learn characteristic descriptions. For instance, in the ID3 algorithm each leaf of the decision tree was augmented with a subtree in order to impose further restrictions on the feature values. A lower and an upper limit are computed for every feature. These will serve as tests:

if the feature value of the instance to be classified is below the lower limit or above the upper limit for one or more of the features, the instance will be rejected, i.e., regarded as belonging to a novel class, otherwise it will be classified according to the original decision tree. Thus, when a new instance is to be classified, the decision tree is first applied as usual, and then, when a leaf would have been reached, every feature of the instance is checked to see if it belongs to the interval defined by the lower and the upper limits. If all features of the new instance are inside their interval the classification is still valid, otherwise the instance will be rejected. In this method the lower and upper limits were assigned the minimum and maximum feature value of the training instances of the leaf respectively. Since this approach will yield a *maximum specific description* (cf. Holte et al. [HAP89]), we will refer to it as ID3-Max.

The main result of Mårtensson's study was that the performance in terms of classification accuracy of three methods was almost equivalent. However, as it became clear that the company wanted explicit classification rules (e.g., for service and maintenance reasons), the neural network approach was given up. It also became apparent that the method must be able to learn disjunctive concepts (i.e., classes that correspond to more than one cluster of training instances in the feature space). In addition to the fact that the actual appearance of a denomination change now and then (and often both are valid as means of payment), angular coins will be in different positions in the sensor resulting in different values in the diameter, and, as mentioned earlier, the alloy is different in different batches. These irregularities will tend to divide the measurements of one type of coin into two or more separate clusters. This was one of the reasons why the Bayesian classifier approach was given up. Another reason was that it does not provide explicit minimum and maximum limits for each parameter, which is what should be programmed into the decision mechanism of the machine.

Since the decision tree approach (1) by far was the most time-efficient in the classification phase, (2) was reasonable fast in the learning phase, (3) learned explicit rules (and provides explicit min and max limits for each parameter), and (4) was good at learning disjunctive concepts, it was selected as the most promising candidate for solving the problem.

Although the ID3-Max algorithm showed promising results, an improvement was desired for mainly two reasons: Firstly, the ID3-Max algorithm is not sufficiently robust, e.g., an extremely low (or high) value of one parameter of one coin in the training set could ruin the whole decision mechanism. This problem is related to the problem of noisy instances which will be discussed in the last section of this paper. Secondly, and perhaps more important, it is not possible to control the degree of generalization (i.e., the position of the limits). For example, too many coins were rejected as the ID3-Max algorithm did not generalize sufficiently. For these reasons, a more dynamic and robust method for computing the limits has been developed.

4 THE NOVEL LEARNING TECHNIQUE

The novel technique developed to solve these problems is also based on the ID3 algorithm and constructs subtrees out of lower and upper limits in a way similar to ID3-Max. The central idea of the method is to make use of statistical information concerning the distribution of the feature values of the instances in the leaf. For every feature we compute the lower and the upper limits so that the probability that a particular feature value (of an instance belonging to this leaf) belongs to the interval between these limits is $1 - \alpha$.

In this way we can control the degree of generalization by choosing an appropriate α-value. The lesser the α-value is, the more will the algorithm generalize. For instance, if it is important not to misclassify instances and a high number of rejected (not classified) instances are acceptable, a high α-value should be selected.

If X is normally distributed stochastic variable, we have that:

$$P(m - \lambda_{\frac{\alpha}{2}}\sigma < x < m + \lambda_{\frac{\alpha}{2}}\sigma) = 1 - \alpha$$

where m is the mean, σ is the standard deviation, and some common values of λ are:

$$\lambda_{0.05} = 1.6449, \quad \lambda_{0.025} = 1.9600, \quad \lambda_{0.005} = 2.5758$$

Thus, we have, for instance, that the probability of an observation being larger than $m - 1.96\sigma$ and smaller than $m + 1.96\sigma$ is 95%.

In order to follow this line of argument we have to assume that the feature values of each category (or each leaf if it is a disjunctive concept) are normally distributed. This assumption seems not too strong for most applications. However, as we typically cannot assume that the actual values of m and σ are known, they have to be estimated. A simple way of doing this is to compute the mean and the standard deviation of the training instances (belonging to the current leaf):

$$m^* = \frac{\Sigma x_i}{n}, \quad \sigma^* = \sqrt{\frac{\Sigma(x_i - \overline{x})^2}{n-1}}$$

To get a nice interpretation of the interval between the upper and lower limit, we have to assume that these estimates are equal to the actual values of m and σ. This is, of course, too optimistic, but it seems reasonable to believe (and will be shown in Section 6) that the method is of practical value also without this interpretation. Anyway, the intended statistical interpretation suggests that the probability of a feature of an instance of a category being larger than lower limit and smaller than upper limit for $\alpha = 0.01$ is 99%.

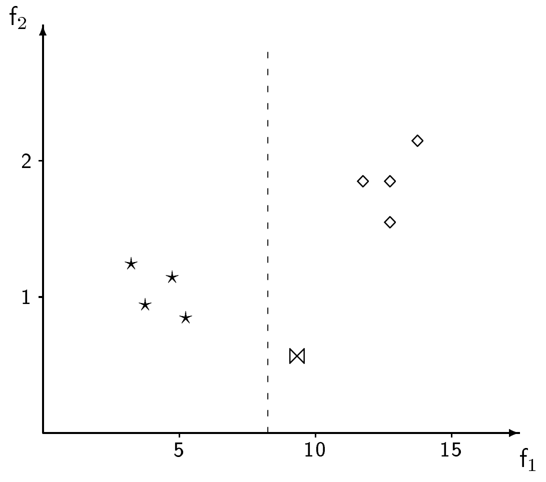

Figure 2: The feature space of the example. The boundary between the two categories ($\star$ and $\diamond$) induced by the ID3 algorithm is represented by the vertical dashed line.

5 A SIMPLE EXAMPLE

We will now present a very simple example to illustrate the method. All instances are described by two numerical features, and the training instances belong to either of two categories: the $\star$-category or the $\diamond$-category. The system is given four training instances of each category.

The feature values of the training instances of the $\star$-category are: (3.0, 1.2), (3.5, 0.9), (4.5, 1.1), (5.0, 0.8) and the $\diamond$-category training instances are: (11.5, 1.8), (12.5, 1.5), (12.5, 1.8), (13.5, 2.1). Figure 2 shows the positions of the instances in the feature space.

If these training-instances are given to the ID3 algorithm, the output will be the decision tree shown in Figure 3. (Or a similar one, depending on the cut-point selection strategy. In all examples presented here the cut-point is chosen by first sorting all values of the training instances belonging to the current node. The cut-point is then defined as the average of two consecutive values of the sorted list if they belong to instances of different classes.) This tree represents the decision rule: if $f_1 \leq 8.25$ then the instance belongs to the $\star$-category, else it belongs to the $\diamond$-category. The classification boundary that follows from this rule is illustrated in Figure 2 by a vertical dashed line. If we now apply the decision tree to an instance of another category ($\bowtie$) with the feature values (9.0,0.5), it will be (mis)classified as an instance of the $\diamond$-category.

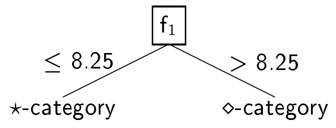

Figure 3: The decision tree induced by the ID3 algorithm.

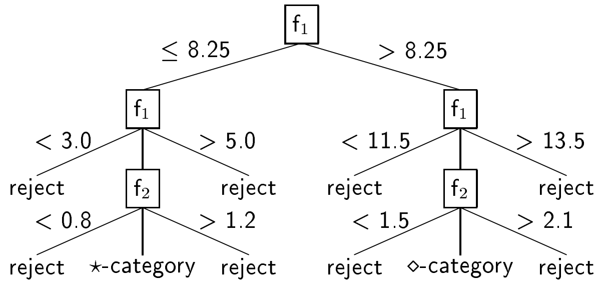

Figure 4: The decision tree induced by ID3-Max.

Let us apply the method based on the maximum specific description to this problem. Given the training instances it will produce the decision tree in Figure 4 which will reject all instances outside the dotted boxes in Figure 5.

If we now apply the augmented decision tree to the $\bowtie$-category instance, we first use the decision tree as before resulting in a preliminary classification which, still as before, suggests that it belongs to the $\diamond$-category. However, as we proceed further down the tree into the appended subtree, we will eventually encounter a test that brings us to a reject-leaf (i.e., we check whether the new instance is inside the dotted box, and find out that it is not). As a consequence, the instance is rejected and treated as an instance of a novel, or unknown, category.

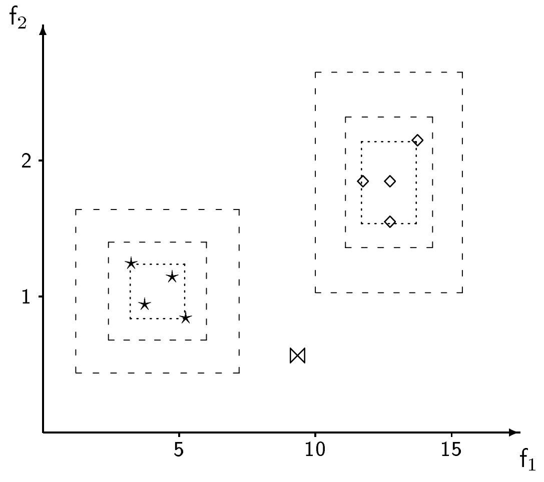

Figure 5: The acceptance regions of the maximum specific description (dotted boxes) and those resulting from the method based on statistical distribution with $\alpha = 0.05$ (inner dashed boxes), and with $\alpha = 0.001$ (outer dashed boxes).

	Canadian Coins			Foreign Coins		
	correct	miss	reject	correct	miss	reject
ID3	99.7%	0.3%	0.0%	0.0%	100.0%	0.0%
ID3-Max	83.7%	0.0%	16.3%	0.0%	0.0%	100.0%
ID3-SD 0.1	62.1%	0.0%	37.9%	0.0%	0.0%	100.0%
ID3-SD 0.05	77.5%	0.0%	22.5%	0.0%	0.0%	100.0%
ID3-SD 0.01	92.2%	0.0%	7.8%	0.0%	0.0%	100.0%
ID3-SD 0.001	97.9%	0.0%	2.1%	0.0%	0.0%	100.0%
ID3-SD 0.0001	98.9%	0.0%	1.1%	0.0%	0.0%	100.0%
desired	100.0%	0.0%	0.0%	0.0%	0.0%	100.0%

Table 1: Result from training set containing Canadian coins (averages over 10 runs).

If we apply the method based on statistical distribution with α = 0.05, the lower and upper limits will have the following values: for the $\star$-category 2.2 (f_1 lower), 5.8 (f_1 upper), 0.6 (f_2 lower), and 1.4 (f_2 upper), and for the $\diamond$-category 10.9, 14.1, 1.3, and 2.3 respectively. These limits will yield a decision tree with the same structure as that of the maximum specific method but with different values on the rejection branches, and will cover the areas marked by the inner dashed boxes in Figure 5. Such an area, which we will call *acceptance region*, can be interpreted as meaning that, if the assumptions mentioned above were correct and if the features are independent, 90.2% (0.95$\times$0.95) of the instances of the category are inside the box. Just as with the maximum specific tree this tree will reject the $\bowtie$-category instance. We can also see that the lesser α-value that is chosen, the more will the algorithm generalize. The outer dashed boxes correspond to the acceptance region for α = 0.001, i.e., 99.8% of all instances of the category are inside the region.

6 TEST RESULTS

In our experiments two databases were used, one describing Canadian coins contains 7 categories (1, 5, 10, 25, 50 cent, 1 and 2 dollar), and one describing Hong Kong coins that also contains 7 categories (5, 10, 20, 50 cent, 1, 2, and 5 dollar). All of the 5 attributes (diameter, thickness, conductivity1, conductivity2, and permeability) are numerical. The Canada and Hong Kong databases were chosen because when using the company's current method for creating the rules of the decision mechanism, these coins have been causing problems.

In each experiment 140 (7$\times$20) instances were randomly chosen for training and 700 (2$\times$7$\times$50) instances for testing. This scenario is quite similar to the actual situation where you in the training phase expose the system only to the coins of one country, but in the classification phase also confront it with coins of other countries. Each experiment was performed with the original ID3 algorithm, the maximum specific tree algorithm (ID3-Max), and the algorithm based on statistical distribution (ID3-SD) for the α-values: 0.1, 0.05, 0.01, 0.001, and 0.0001.

Table 1 shows the classification results when training on the Canadian coin database. We can see that all foreign coins (i.e., Hong Kong coins) are rejected, except of course for the ID3 algorithm. Neither were there any problems with misclassifications. In fact, the Canadian coins that are misclassified by the ID3 algorithm are rejected by the algorithms learning characteristic descriptions. However, the requirements of classification accuracy (less than 5% rejects of known types of coins and very few misclassifications (less than 0.5%)) are met only by the ID3-SD algorithm with α = 0.001 and 0.0001, which illustrates the advantage of being able to control the degree of generalization.

In Table 2 the results when training on the Hong Kong coin database are shown. As indicated by the percentages of misclassifications of known types of coins, this is a more difficult problem. Although there are two α-values (0.001 and 0.0001) that meet the requirements, they are very close to the acceptable number of rejects (0.001) and misclassifications (0.0001) respectively. Results from experiments using other databases can be found in Davidsson [Dav95].

A potential problem for the ID3-SD algorithm is when the training set consists of only a few training instances of one or more of the categories (or clusters). One would think that when the number of training examples of a category decreases there is a risk that the estimates of the mean value and the standard deviation (which are fundamental for computing the acceptance regions) will not be sufficiently good. However, preliminary experiments indicate that the classification performance decreases only slowly when the training

	Hong Kong Coins			Foreign Coins		
	correct	miss	reject	correct	miss	reject
ID3	98.3%	1.7%	0.0%	0.0%	100.0%	0.0%
ID3-Max	79.7%	0.0%	20.3%	0.0%	0.0%	100.0%
ID3-SD 0.1	60.0%	0.0%	40.0%	0.0%	0.0%	100.0%
ID3-SD 0.05	74.8%	0.0%	25.2%	0.0%	0.0%	100.0%
ID3-SD 0.01	88.9%	0.0%	11.1%	0.0%	0.0%	100.0%
ID3-SD 0.001	95.1%	0.3%	4.6%	0.0%	0.0%	100.0%
ID3-SD 0.0001	96.3%	0.5%	3.2%	0.0%	0.0%	100.0%
desired	100.0%	0.0%	0.0%	0.0%	0.0%	100.0%

Table 2: Result from training set containing Hong Kong coins (averages over 10 runs).

examples get fewer. As can be seen in figure 6, it handles the problem of few training instances much better than the maximum specific description approach which, in fact, has been suggested as a solution to the related problem of small disjuncts (cf. Holte et al. [HAP89]). In the coin classification domain small disjuncts could, for instance, be caused by irregularities in the composition of the alloys.

7 NOISY AND IRRELEVANT FEATURES

The ID3-SD algorithm is better at handling noisy data than the ID3-Max algorithm in the sense that an extreme feature value for one (or a few) instance(s) will not influence the positions of the limits of that feature in ID3-SD as much as it will in ID3-Max. This is illustrated in Figure 7 where a single instance with a single noisy feature value corrupts the acceptance region of the Max-algorithm whereas the acceptance region of the SD-algorithm is affected to a lesser extent.

A method for further reducing the problem of noisy instances, would be to use the acceptance regions to remove instances that are (far) outside their acceptance region and then recalculate the region. For instance, if we remove the noisy instance in the figure and recalculate the acceptance region, we get the region shown in the right picture in the figure. This method for *outlier detection* is currently under evaluation.

However, in this paper we have used the traditional ID3 algorithm as a basis, an algorithm that is not very good at handling noisy data in the first place. In fact, there is a triv-

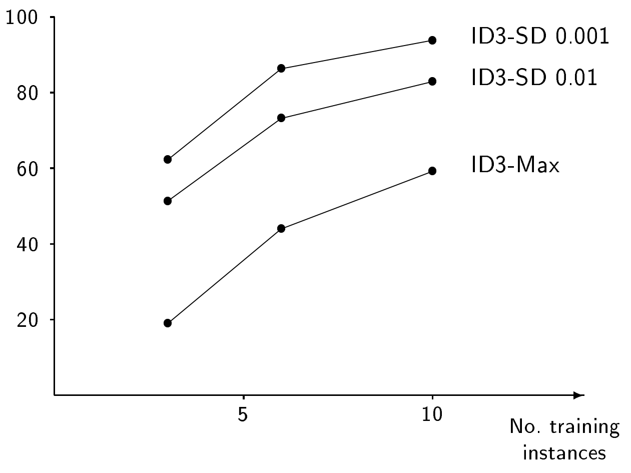

Figure 6: The percentage of correctly classified instances of known categories (Canadian coins) as a function of the number of instances of each category in small training sets (averages over 10 runs). The remaining instances were rejected.

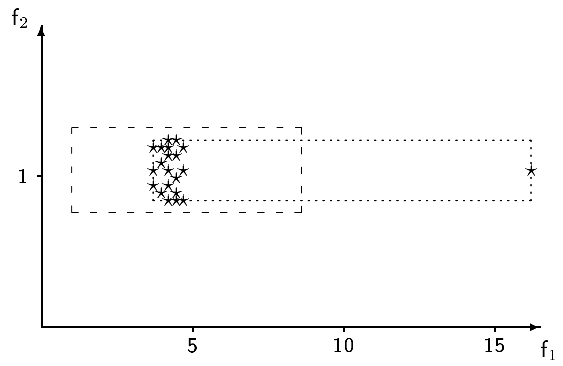

Figure 7: A category with twenty good and one noisy instance. The maximum specific description correspond to the dotted box. The dashed box correspond to the description resulting from the ID3-SD algorithm with $\alpha = 0.1$.

ial solution to the problem with noisy data: Use a pruning method (cf. Mingers [Min89]) to cut off the undesired branches, or use any other noise tolerant algorithm (e.g., C4.5 [Qui93]) for inducing decision trees, and then compute the subtrees as before for the remaining leaves.

Moreover, the original ID3-algorithm is quite good at handling the problem of irrelevant features (only features that are useful for discriminating between the categories in the training set are selected). But since the suggested method computes upper and lower limits for every feature and use these in the classification phase, also the irrelevant features will be subject for consideration. However, this potential problem will typically disappear since an irrelevant feature often is defined as a feature which value is randomly selected according to a uniform distribution on the feature's value range (cf. Aha [Aha92]). That is, the feature values have a large standard deviation, which will lead to a large gap between the lower and the upper limit. Thus, as most values of the feature will be inside the acceptance region with regard to this feature, the feature will still be irrelevant for the classification.

8 THE GENERALITY OF THE SD APPROACH

The main limitation of the SD approach seems to be that it is only applicable to numerical attributes. The maximum specific description method, on the other hand, requires only that the features can be ordered. Thus, one way of making the former method more general is to combine it with the latter method to form a hybrid approach that is able to handle all kinds of ordered features. We would then use the statistical method for numerical attributes and the maximum specific description method for the rest of the attributes. Moreover, nominal attributes could be handled by accepting those values present among the instances of the leaf and reject those that are not. In this way we get a method that learns characteristic descriptions using all kinds of attributes. However, the degree of generalization can, of course, only be controlled for numeric features.

The statistically based approach for creating characteristic descriptions is a general method in the sense that we can take the output from any decision tree induction algorithm, compute a subtree for every leaf, and append them to their leaf. In fact, the approach can, in principle, be applied to any empirical learning method, supervised or unsupervised, using the hybrid approach. However, if the instances of a category corresponds to more than one cluster in the feature space (cf. disjunctive concepts), the method will probably work better for algorithms that explicitly separates the clusters, i.e., where it is possible to find out which cluster a particular instance belongs to. If this is the case, the acceptance regions can be computed separately for each cluster. Otherwise, we must compute only one acceptance region for the

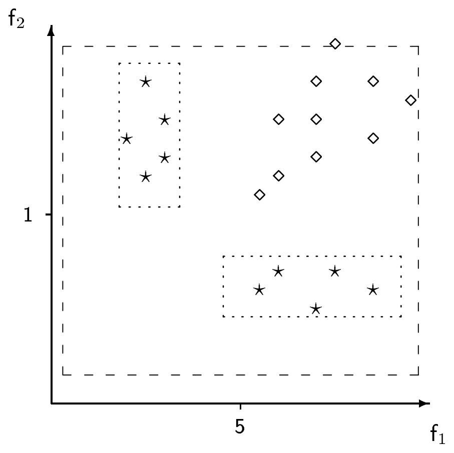

Figure 8: The acceptance regions for the $\star$-category computed by ID3-SD (dotted boxes) and by an algorithm that does not explicitly separate clusters of category instances (dashed box). ($\alpha = 0.05$).

whole category, which probably will result in a too large acceptance regions (see Figure 8).

The problem with disjunctive categories is also a partial answer to the question: Why bother building a decision tree in the first place? Could we not just compute the lower and the upper limits for every category and test unknown instances against these? The main problem with such an approach would be that when a new instance is to be classified, it might be assigned to two or more classes. The reason for this ambiguity is, of course, that the acceptance regions of two or more categories may overlap. Moreover, even if the regions do not overlap, there will be problems dealing with disjunctive concepts for the reason mentioned above. Thus, in either case, we must have an algorithm that is able to find suitable disjuncts of the concept (which, in fact, is an unsupervised learning problem), a task that ID3-like algorithms normally are quite good at. However, Van de Merckt [dM93] has suggested that for numerical attributes a similarity-based selection measure is more appropriate for finding the correct disjuncts than the original entropy-based measure used in the empirical evaluations presented here.

The procedure for augmenting an arbitrary empirical learning algorithm X is as follows: train X as usual, then compute the limits for every category (i.e., cluster) in the training set as described earlier. When a new instance is to be classified, first apply X's classification mechanism in the same way as usual, then check that all features values of the new instance are larger than the lower limit and smaller than the upper limit. Thus, it is not necessary to represent the limits in the form of decision trees, the main point is that there should be a method for comparing the feature values of the instance to be classified with the limits.

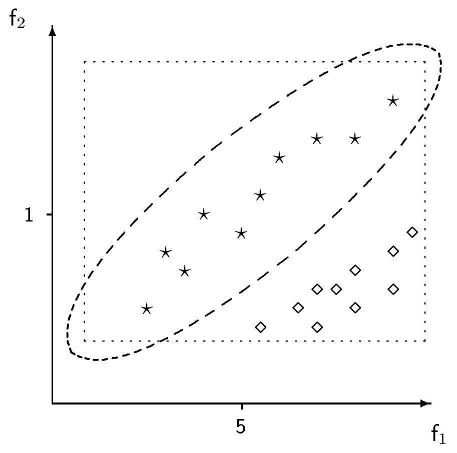

Figure 9: The acceptance regions for the ⋆-category computed by IB1-SD (dotted boxes) and by an algorithm that takes into account covariation among features (dashed ellipse). ($\alpha = 0.05$).

To illustrate the generality of the SD approach some experiments with its application to a nearest neighbour algorithm very similar to IB1 [AKA91] are described in Davidsson [Dav96]. Moreover, if we are not restricted to representing concepts by decision trees, we are not forced to have separate and explicit limits for each feature. As a consequence, we do not have to assume that features are independent. If we are able to capture covariation among two or more features we would be able to create acceptance regions that closer match the distribution of feature values (i.e., regions that are smaller but still cover as many instances). This is illustrated by the example shown in Figure 9. The acceptance region for the ⋆-category computed by IB1-SD does not fit the training instances very well and does actually also cover all training instances of the ⋄-category despite that the members of the two categories forms clearly separated clusters. The acceptance region computed by the algorithm able to capture covariances between features, on the other hand, do not cover any of the ⋄-instances.

To implement the latter algorithm it is necessary to apply multivariate statistical methods. In particular the following theorem is useful: assuming that feature values are normally distributed within categories/clusters we have that the solid ellipsoid of x values satisfying

$$(x-\mu)^T\Sigma^{-1}(x-\mu) \leq \chi^2_p(\alpha)$$

has probability $1-\alpha$, where μ is the mean vector, Σ is the covariance matrix, χ^2 is the chi-square distribution and p is the number features. (See for instance [JW92].) This theorem can be used to compute a weighted distance from the instance to be classified to the "center" of the category/cluster. If this distance is larger than a critical value (dependent of α) the instance is rejected. Thus, instead of estimating m and σ, we have to estimate μ and Σ for each category/cluster from the training set. An algorithm based on this idea together with some very promising experimental results are presented in Davidsson [Dav96].

Unfortunately, such an algorithm is not applicable to the coin sorting problem described above. There are two main reasons for this: (i) the algorithm does not provide explicit min and max limits for each feature, and (ii) the algorithm is too slow in the classification phase; first, the instance to be classified has to be compared to a large number of the training instances, and then, a number of time consuming matrix calculations are necessary in order to compute the weighted distance.

9 CONCLUSIONS

We have applied a novel method for learning characteristic decision trees, the ID3-SD algorithm, to the problem of learning the decision mechanism of coin-sorting machines. The main reason for the success of this algorithm in this application was its ability to control the degree of generalization (an ability which, of course, requires the learning of characteristic descriptions). To author's knowledge, ID3-SD is the first algorithm in which this can be done both explicitly (i.e., by specifying a parameter, the α-value) and without making any ad-hoc assumptions (i.e., it is based on sound statistical reasoning).

In our experiments α-values about 0.0001 has given the best results, but we do not know whether this holds generally. Development of methods to determine automatically the appropriate degree of generalization belongs to future research. Future work will also include an evaluation of how other empirical learning methods can be improved by the SD approach. In this perspective, we have in this paper only described an application of the general method to the ID-3 algorithm (i.e., it can be regarded a case study).

ACKNOWLEDGEMENTS

I wish to thank Eric Astor, Robert M. Colomb, Paul Compton, Len Hamey, Måns Holgersson, and Anders Holtsberg for fruitful discussions and helpful suggestions.

REFERENCES

[Aha92] D.W. Aha. Generalizing from case studies: A case study. In *Ninth International Workshop on Machine Learning*, pages 1–10. Morgan Kaufmann, 1992.

[AKA91] D.W. Aha, D. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[Dav95] P. Davidsson. ID3-SD: An algorithm for learning characteristic decision trees by controlling the degree of generalization. Technical Report LU–CS–TR: 95–145, Dept. of Computer Science, Lund University, Lund, Sweden, 1995.

[Dav96] P. Davidsson. *Autonomous Agents and the Concept of Concepts*. PhD thesis, Department of Computer Science, Lund University, Sweden, 1996. (In preparation.).

[dM93] T. Van de Merckt. Decision trees in numerical attribute spaces. In *IJCAI-93*, pages 1016–1021. Morgan Kaufmann, 1993.

[HAP89] R.C. Holte, L.E. Acker, and B.W. Porter. Concept learning and the problem of small disjuncts. In *IJCAI-89*, pages 813–818. Morgan Kaufmann, 1989.

[Hut94] A. Hutchinson. *Algorithmic Learning*. Clarendon Press, 1994.

[JW92] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice-Hall, 1992.

[Mar94] E. Mårtensson. Improved coin classification, (Master's thesis). LU–CS–EX: 94–2, Dept. of Computer Science, Lund University, Sweden, 1994. (In Swedish).

[Min89] J. Mingers. An empirical comparision of pruning methods for decision tree induction. *Machine Learning*, 4(2):227–243, 1989.

[Qui86] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Qui93] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[RHW86] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol.1: Foundations*. MIT Press, 1986.

[SM92] P. Smyth and J. Mellstrom. Detecting novel classes with applications to fault diagnosis. In *Ninth International Workshop on Machine Learning*, pages 416–425. Morgan Kaufmann, 1992.

[TG74] J.T. Tou and R.C. Gonzales. *Pattern Recognition Principles*. Addison-Wesley, 1974.