Theory and Applications of Attribute Decomposition

Lior Rokach
Department of Industrial Engineering
Tel-Aviv University
Ramat Aviv, Tel Aviv 69978, Israel
liorr@eng.tau.ac.il

Oded Mainon
Department of Industrial Engineering
Tel-Aviv University
Ramat Aviv, Tel Aviv 69978, Israel
maimon@eng.tau.ac.il

Abstract

This paper examines the Attribute Decomposition Approach with simple Bayesian combination for dealing with classification problems that contain high number of attributes and moderate numbers of records. According to the Attribute Decomposition Approach, the set of input attributes is automatically decomposed into several subsets. A classification model is built for each subset, then all the models are combined using simple Bayesian combination. This paper presents theoretical and practical foundation for the Attribute Decomposition Approach. A greedy procedure, called D-IFN, is developed to decompose the input attributes set into subsets and build a classification model for each subset separately. The results achieved in the empirical comparison testing with well-known classification methods (like C4.5) indicate the superiority of the decomposition approach.

1. Introduction

Classification modeling is one of the most important techniques for data mining. In classification problem the induction algorithm is given a set of training instances and the corresponding class labels and outputs a classification model. The classification model takes an unlabeled instance and predicts its class.

Fayyad *et al.* [8] claim that the explicit challenges for the KDD research community is to develop methods that facilitate the use of data mining algorithms for real-world databases. One of the characteristics of a real world databases is high dimensionality. High Dimensionality increases the size of the search space in an exponential manner, and thus increases the chance that the algorithm will find spurious models that are not valid in general. Elder and Pregibon [7] define this phenomenon as the ”curse of dimensionality”. Techniques that are efficient in low dimensions fail to provide meaningful results when the number of dimensions goes beyond a 'modest' size of 10 attributes. Furthermore smaller data mining models, involving less attributes, are much more understandable by humans. Smaller models are also more appropriate for user-driven data mining, based on visualization techniques.

Most of the methods for dealing with high dimensionality focus on *Feature Selection* techniques, i.e. selecting some subset of attributes upon which the induction algorithm will run, while ignoring the rest. In the last decade, *Feature Selection* has enjoyed increased interest by the data mining community. Consequently many *Feature Selection* algorithms have been proposed, some of which have reported remarkable accuracy improvement. The literature on this subject is too wide to survey here, however, we recommend [12] on this topic.

Despite its popularity, using *Feature Selection* methodology for overcoming the high dimensionality obstacles has several shortcomings:

- The assumption that a large set of input attributes can be reduced to a small subset of relevant attributes is not always true; in some cases the target attribute is actually affected by most of the input attributes, and removing attributes will cause a significant loss of important information.

- The outcome of many algorithms for *Feature Selection* is strongly dependent on the training set size. That is, if the training set is small the size of the reduced subset will be small as well.

- In some cases, even after eliminating a set of irrelevant attributes, the researcher is left with relatively large numbers of relevant attributes.

- The backward elimination strategy, used by some methods, is extremely inefficient for working with large-scale databases, where the number of original attributes is large.

A number of linear dimension reducers have been developed over the years. The linear methods of dimensionality reduction include projection pursuit [9], factor analysis [10], and principal components analysis [6]. These methods are not aimed directly at eliminating irrelevant and redundant attributes, but are rather concerned with transforming the observed variables into a small number of "projections" or "dimensions". The underlying assumptions are that the variables are numeric and the dimensions can be expressed as linear combinations of the observed variables. Each discovered dimension is assumed to represent an unobserved factor and thus provide a new way of understanding the data.

The linear dimension reducers have been enhanced constructive induction systems that use a set of existing attributes and a set of predefined constructive operators to derive new attributes [17]. Zupan *et al.* [24] presented a general-purpose function decomposition approach for machine learning. According to this approach, attributes are transformed into new concepts in an iterative manner and create a hierarchy of concepts. These methods are effective for high dimensionality applications only if the original domain size of the input attribute can be in fact decreased dramatically.

One way to deal with the aforementioned disadvantages is to use a very large training set (which should grows in an exponential manner as the number of input attributes increases). However, the researcher rarely enjoys this privilege, and even if it does happen, the researcher will probably encounter the aforementioned difficulties derived from high number of records.

The rest of the paper is organized into three parts. In the first part we introduce the Attribute Decomposition Approach literally and theoretically. In the second part we develop a heuristic algorithm for implementing the decomposition approach. In the third part we examine the algorithm on several artificial data and real applications.

2. Attribute Decomposition Approach

The purpose of decomposition is to break down a complex problem into several manageable problems.

Problem decomposition's benefits include: conceptual simplification of the problem, making the problem more feasible by reducing its dimensionality, achieving clearer results, reducing run time by solving smaller problems and by using parallel computation, and allowing different solution techniques for individual sub problems.

The decomposition approach is frequently used for operation research and engineering design, however, as Buntine [3] states, it has not attracted as much attention in KDD and machine learning.

In this paper we present the attribute decomposition approach where the original attribute set is decomposed to mutually exclusive subsets by an algorithm. A learning algorithm is run upon the training data for each subset independently and then the generated models are combined in order to classify new instances. This method facilitates the creation of a classification model for high dimensionality databases.

Variety of methods (like boosting or bagging), which also provide an improvement in classification performance by combining several simple classifiers produced by the same method, can be found in the literature. However, the resulting predictions are usually inscrutable to end-users [19].

Other methods that deals directly with high dimensional data like support vector machines also suffer from inscrutability. The method proposed here improves classification performance, without jeopardizing the model's comprehensibility.

The problem of decomposing the input attributes set is that of finding the best decomposition, such that if a specific induction algorithm is run on each attribute subset data, then the combination of the generated classifiers will have the highest possible accuracy.

In this paper we focus on simple Bayesian combination, which is an extension of the *Simple Bayesian* classifier. The *Simple Bayesian* classifier uses Bayes rule to compute the probability of each possible value of the target attribute given the instance, assuming the input attributes are conditionally independent given the target attribute. The predicted value of the target attribute is the one, which maximizes the calculated probability. Due to the attribute independence assumption, this method is also known as "naive" Bayesian. However, a variety of empirical research shows surprisingly that the simple Bayesian classifier can perform quite well compared to other methods even in domains where clear attribute dependencies exist. Furthermore, simple Bayesian classifiers are also very simple and easy to understand.

In the attribute decomposition approach with simple Bayesian combination we use a similar idea, namely the prediction of a new instance s_q is based on the product of the conditional probability of the target attribute, given the values of the input attributes in each subset. Mathematically it can be formulated as follows:

$$v_{MAP} = \underset{v_{y,j} \in V_y}{\operatorname{argmax}} \frac{\prod_{k=1}^{\omega} \hat{P}_I(y=v_{y,j} | a_i=x_{q,i} \ i \in R_k)}{\hat{P}_I(y=v_{y,j})^{\omega-1}}$$

Where $x_{q,i}$ denotes the value of attribute i in an observation q. V_y represents the domain of the target attribute. R_k denotes the indexes of the attributes that belong to subset k. y represents the class variable or the target attribute.

In fact extending the simple Bayesian classifier by joining attributes is not new. Kononenko [11] used a conditional independence test to join attributes. Domingos and Pazzani [4] used estimated accuracy (as determined by leave one out cross validation on the training set). In both cases, the suggested algorithm finds the single best pair of attributes to join by considering all possible joins. However these methods have not noticeably improved accuracy. The reasons for the moderate results are two-fold. First both algorithms used a limited criterion for joining attributes. Second and more importantly, attributes are joined by creating a new attribute, whose values are the Cartesian product of its ingredients' values, specifically the number of attributes that can be joined together is restricted to a small number. Furthermore the problem have not been formally defined and explored.

Duda and Hart [5] showed that Bayesian classifier has highest possible accuracy (i.e. Bayesian classifier predicts the most probable class of a given instance based on the complete distribution). However in practical learning scenarios, where the training set is very small compared to the whole space, the complete distribution can hardly be estimated directly.

According to the decomposition concept the complete distribution is estimated by combining several partial distributions. Bear in mind that it is easier to build a simpler model from limited number of training instances because it has less parameters. This makes classification problems with many input attributes more feasible.

This problem can be related to the bias-variance tradeoff. The bias of a learning algorithm for a given learning problem is the persistent or systematic error that the learning algorithm is expected to make. A concept closely related to bias is variance. The variance captures random variation in the algorithm from one training set to another. This variation can result from variation in the training sample, from random noise in the training data, or from random behavior in the learning algorithm itself. The smaller each subset is, the less probabilities we have to estimate and potentially less variance in the estimation of each one of them. On the other hand, when there are more subsets, we expect that the approximation of the full distribution using the partial distributions is less accurate (i.e. higher bias error). Formally the problem can be phrased as follows:

Given a learning method I, and a training set S with input attribute set $A = \{a_1, a_2, ..., a_n\}$ *and target attribute y from a distribution D over the labeled instance space, the goal is to find an optimal decomposition* Z_{opt} *of the input attribute set A into* ω *subsets* $G_k = \{a_{k,j(i)} | j = 1, ..., l_k\}$ $k = 1, ..., \omega$ *fulfilling* $(\bigcup_{k=1}^{\omega} G_k) \subseteq A$ *and* $G_i \cap G_j = \emptyset; i, j = 1, ..., \omega; i \neq j$ *such that the generalization error of the simple Bayesian combination of the induced classifiers will be minimized over the distribution D.*

It should be noted that the optimal is not necessarily unique. Furthermore the problem can be treated as an extension of the feature subset selection problem, i.e. finding the optimal decomposition of the form $Z_{opt} = \{G_1\}$, as the non-relevant features are in fact $A - G_1$. Moreover, the Naive Bayes method can be treated as specific decomposition where each subset contains a single attribute.

Definition 1 (Complete Equivalence) *The decomposition* $Z = \{G_1, ...G_k..., G_\omega\}$ *is said to be completely equivalent if for each instance with positive probability* ($\forall x_q \in X; P(x_q) > 0$)*, the following is satisfied:*

$$\underset{v_{y,j} \in V_y}{\operatorname{argmax}} \frac{\prod_{k=1}^{\omega} P(y = v_{y,j} | a_i = x_{q,i} \; i \in R_k)}{P(y = v_{y,j})^{\omega - 1}} = \underset{v_{y,j} \in V_y}{\operatorname{argmax}} P(y = v_{y,j} | a_i = x_{q,i} \; \forall a_i \in A)$$

Since Duda and Hart showed that the right term of the equation is optimal, it follows that a completely equivalent decomposition is optimal as well. The importance of finding a completely equivalent decomposition is derived from the fact that in real problems with limited training sets, we can only estimate the probability and it is easier to approximate probabilities with lower dimensions.

It can be shown that the number of possible decompositions (i.e. the size of the search space) increases in a strong exponential manner as the number of input attributes grows [14], i.e. an exhaustive search is not practical for high number of input attributes.

Hereby we present three Lemmas that will shed light on the suggested problem. The proof of the following Lemmas is straightforward.

Lemma 2 (Sufficient condition) *Let Z be a decomposition, which satisfies* $G_k, k = 1, ..., \omega$ *and* $NR = A - \bigcup_{k=1}^{\omega} G_k$ *are conditionally independent given y and that NR and y are independent, then Z is completely equivalent.*

The above Lemma represents a sufficient condition for complete equivalence. However it is important to note that it does not represent a necessary condition, the following Lemma illustrates this statement.

Lemma 3 *Let* $A = \{a_1, ..., a_l, ..., a_n\}$ *be a group of n independent input binary attributes and let* $Z = \{G_1, ..., G_\omega\}$ *be a decomposition, then if the target attribute follows the function:*

$$y = f_1(a_i, i \in R_1) \vee f_2(a_i, i \in R_2) \vee ... \vee f_\omega(a_i, i \in R_\omega)$$

or

$$y = f_1(a_i, i \in R_1) \wedge f_2(a_i, i \in R_2) \wedge ... \wedge f_\omega(a_i, i \in R_\omega)$$

Where $f_1, ..., f_\omega$ *are Boolean functions and* $R_1, ..., R_\omega$ *are mutually exclusive then Z is completely equivalent.*

The above Lemma illustrates that although the conditionally independent requirement is not fulfilled we still find a completely equivalent decomposition.

Lemma 4 (XOR Problem) *Let $A = \{a_1, ..., a_i, ..., a_n\}$ be a group of n input binary attributes, if the target attribute behave as $y = a_1 \oplus a_2 \oplus ... \oplus a_n$ then there is no decomposition beside $Z = \{A\}$ which is completely equivalent.*

The last Lemma shows that there are problems that no completely equivalent decomposition can be found, beside the obvious one.

3. D-IFN Algorithm

Maimon and Last [13] present a *Multi-Layer Info-Fuzzy Network (IFN)* aimed at finding the minimum number of input attributes required for predicting a target attribute. Each vertical layer is uniquely associated with an input attribute by representing the interaction of that attribute and the input attributes of the previous layers. The first layer includes only the root node and is not associated with any input attribute. This structure is also known as *Oblivious Decision Trees*. The multi-layer network can also be used to predict values of target attributes in a disjunctive manner, similar to the decision tree. The principal difference between the structure of a multi-layer network and a decision-tree structure is the *constant ordering* of input attributes at every predicting node of the multi-layer network, the property which is necessary for minimizing the overall subset of input attributes (resulting in dimensionality reduction).

Figure 1 represents a typical IFN with three input attributes: glucose level, age and blood pressure of a certain patient and the Boolean target attribute representing whether that patient suffer from diabetes. The arcs that connect the terminal nodes and the nodes of the target layer are labeled with the number of records that fit this path. For instance there are twelve patients in the training set whose their glucose level is less than 107 but who still suffer from diabetes. The in-depth description of the IFN Methodology can be found in a new book by Maimon and Last (2000).

As the IFN method was found to be effective in discovering the relevant attributes and their relations to the target attribute, we further extended it aiming to approach the optimal decomposition. For our purpose each subset is represented by a different network, while each attribute in this subset is located in a different layer. Attributes in the same network should be dependent, namely independent attributes should be in different networks as the independence assumption suggests. However it does not mean that attributes in different networks are necessarily independent. In some cases assigning dependent attributes into different groups contributes to the overall prediction accuracy.

Obviously we can construct up to n networks (in the extreme case where each network may represent one input attribute) and up to n attributes in each network (in the case where we have one network that include all input attributes).

For creating the multiple networks we use a greedy depth-first algorithm, called D-IFN. The D-IFN learning algorithm starts with a single network with a single node (the root node), representing an empty subset of input attributes. In each iteration the algorithm decides which attribute should be added to the current network as a new layer, and to what nodes on the previous layer it will be connected (the splitted nodes). The nodes of a new layer are defined as all Cartesian product combinations of the splitted nodes with the values of the new input attribute, which have at least one observation associated with it.

The selection of the a new input attribute is made according the following criteria:

- The selected attribute should maximize the total significant decrease in the conditional entropy, as a result of adding it as a new layer. In order to calculate the total significant decrease in the conditional entropy, we estimate for each node in the last layer, the decrease in the conditional entropy as a result of splitting that node according the candidate attribute values. Furthermore the decrease in the conditional entropy is tested for significance by a likelihood-ratio test [2]. The null hypothesis (H_0) is that a candidate input attribute and a target attribute are conditionally independent, given the node (implying that their conditional mutual information is zero). If H_0 holds, the test statistic is distributed as χ^2 with degrees of freedom equal to the number of independent proportions estimated from the observations associated with this node. Finally all significant decreases of specific nodes are summed up to achieve the total significant decrease.

- The attribute is conditionally dependent on the splitted nodes given the target attribute. For testing the conditional independency we use a standard test [23]. The null hypothesis (H_0) is that a candidate input attribute and all splitted nodes are conditionally independent, given the target attribute.

- Adding the attribute to the current network should decrease the generalization error bound of the combined networks so far. The bound is discussed in the following section. In this case there are two purposes for using it; First of all as adding a new attribute to the current network increases its complexity. Under the Occam's-razor assumption that simplicity is a virtue in conflict with training accuracy, we verify whether the decrease in the entropy worth the addition in the complexity; Moreover we check whether the addition of the attribute to the current network, contributes to the performance of the combined networks structure as a whole.

If no input attribute was selected an attempt is made to construct a new network (a new subset) with the input attributes that were not used by previous networks. This procedure continues until there is no more unused attributes or until the last network is left empty. Figure 2 presents the main flowchart of the algorithm.

The networks constructed by D-IFN algorithm, can be used for classification of unlabeled instances, by performing the following steps:

- For each network:
  - Locate the relevant node (final or unsplitted) that satisfies the unseen instance.
  - Extract the frequency vector (how many instances relate to each possible value of the target attribute.)
  - Transform the frequency vector to probability vector. Using the frequency as is will typically over-estimate the probability so we use the Laplace correction to avoid this phenomenon [4].
- Combine the probability vectors using Naive Bayesian combination.
- Select the target value maximizing the Naive Bayesian combination.

The complexity of the proposed algorithm can be bounded by: $O(n^2 \cdot m \cdot d_y \cdot \max(d_i))$

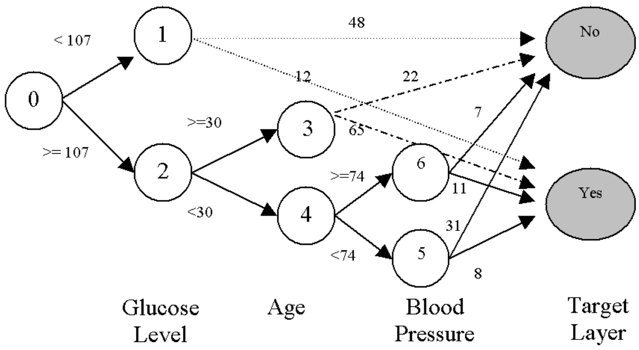

Figure 1. Illustration of IFN.

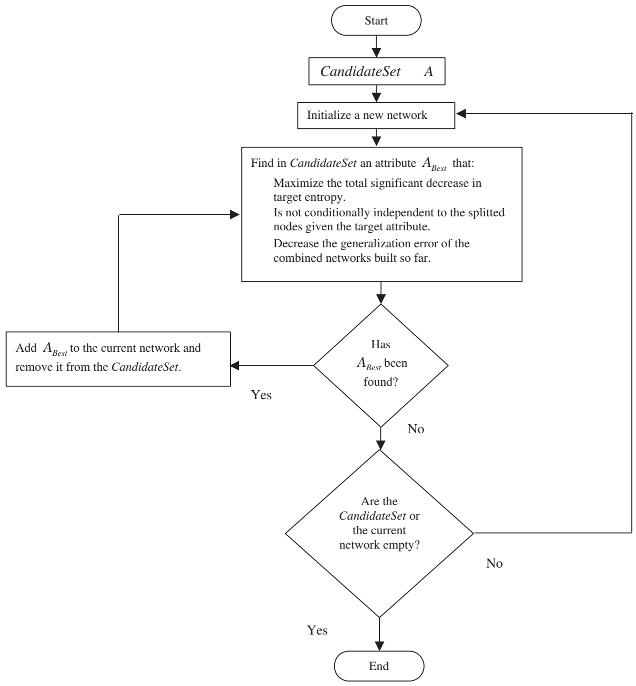

Figure 2. Flowchart of D-IFN Algorithm

4. Generalization Error Bound for D-IFN

Over-fitting and under-fitting are well known problems in learning. A conventional approach to overcome these problems is to use a generalization error bound in terms of the training error and concept size.

As stated before, a IFN can be considered as restricted decision tree, for that reason we might use one of the generalization error bound that were developed for decision trees in the literature [15]. However there are several reasons to develop a specific bound. First, we can utilize the fact that the considered structure is more restricted, in order to develop a tighter bound. Second we are required to extend the bound for several networks using simple Bayesian combination.

Due to the fact that we use probabilistic classifiers (represented by real value weights on the terminal nodes), the hypothesis space is infinite. A typical bound for infinite hypothesis space, which makes use of VC dimension, and holds with probability at least $1-\delta$, has the following form [21]:

$$|\varepsilon(h) - \hat{\varepsilon}(h)| \leq \sqrt{\frac{d \cdot (\ln \frac{2m}{d} + 1) - \ln \frac{\delta}{4}}{m}} \quad \begin{array}{l} \forall h \in H \\ \forall \delta > 0 \end{array}$$

Where d is the VC-Dimension of hypothesis class H, $\hat{\varepsilon}(h)$ is the error of h on training set of cardinality m.

The VC dimension for a set of indicator functions is defined as the maximum number of data points that can be shattered by the set of admissible functions. By definition, a set of m points is shattered by a concept class if there are functions in the class that split the points into two classes in all of the 2^m possible ways. The VC dimension might be difficult to compute accurately. Hereby we introduce an upper and lower bound of the VC dimension.

Theorem 5 *The VC-Dimension of ω mutually exclusive IFNs on n binary input attributes that are combined using simple Bayesian combination and that have $\vec{L} = (l_1, \ldots, l_\omega)$ layers and $\vec{T} = (t_1, ..., t_\omega)$ terminal nodes is not greater than:*

$$\begin{cases} F + \log U & \omega = 1 \\ 2(F+1)\log(2e) + 2\log U & \omega > 1 \end{cases}$$

and at least:

$$F - \omega + 1$$

where:

$$U = \frac{n!}{(n - \sum\limits_{i=1}^{\omega} l_i)!} \cdot \prod_{i=1}^{\omega} \frac{(2t_i - 4)!}{(t_i - 2)! \cdot (t_i - 2)!} \; ; \; F = \sum_{i=1}^{\omega} t_i$$

For proofing the theorem we first consider two Lemmas:

Lemma 6 *The VC dimension of IFN on n binary input attributes with l layers and t terminal nodes is not greater than:*

$$t + \log_2\left(\frac{n!}{(n-l)!} \cdot \frac{(2t-4)!}{(t-2)! \cdot (t-2)!}\right)$$

Proof: Any IFN can be converted to a suitable classification tree with leafs labeled $\{0,1\}$ according to the highest weight of each of the terminal node in the original network. Because the IFN and its corresponded classification tree shattered the same subsets, their VC dimensions are identical.

The hypothesis space size of an IFN with l layers, t terminal nodes and n input attributes to choose from is not greater than:

$$\frac{n!}{(n-l)!} \cdot 2^t \cdot \frac{(2t-4)!}{(t-2)! \cdot (t-2)!}$$

The first multiplier indicates the number of combinations for selecting with order l attributes from n. The second multiplier corresponds to the different classification options of the terminal nodes. The third multiplier represents the number of different binary tree structures that contains t leaves. The last multiplier is calculated using standard tree structure [22].

Based on the familiar relation $VC(H) \leq \log_2(|H|)$ for finite H, we proved the Lemma.

Lemma 7 *Consider ω mutually exclusive IFNs that are combined with simple Bayesian and that have a fixed structure containing $\vec{T} = (t_1, ..., t_\omega)$ terminal nodes. The number of dichotomies it induces on a set of cardinality m is at most:*

$$2\left(em \Big/ (1+F)\right)^{1+F}$$

Proof: For proving the Lemma, we use a similar Lemma introduced by Schmitt [20] for the number of dichotomies that a higher order threshold neuron with k monomials induces on a set of cardinality.

Multiple IFNs which are combined with simple Bayesian, can be converted to a higher order threshold neuron, where the set of terminal nodes constitutes the neuron's monomials and the log-odds in favor of $y = 1$ in each terminal node is the corresponded neuron's weight. Furthermore to be able to use the sign activation function we use a threshold that equals to the sum of all other monomials.

Proof of Theorem 5: We first discuss the upper bound. For a single IFN we can use Lemma 6 directly. For the multiple IFNs we first bound the number of dichotomies induced by ω mutually exclusive IFNs on an arbitrary set of cardinality m. Because the biggest shattered set follows this bound as well, we derive the statement of theorem.

There are at most:

$$U = \frac{n!}{(n - \sum\limits_{i=1}^{\omega} l_i)!} \cdot \prod_{i=1}^{\omega} \frac{(2t_i - 4)!}{(t_i - 2)! \cdot (t_i - 2)!}$$

different structures for ω mutually exclusive IFNs. Combining the last result with Lemma 7 we derive that there are at most:

$$U \cdot 2\left(\frac{em}{F+1}\right)^{F+1}$$

dichotomies on a given set of cardinality m that are induce by the class considered. If the above class shatters the given set, we must have:

$$2^m \leq U \cdot 2\left(\frac{em}{F+1}\right)^{F+1}$$

However the last inequality will not be true if we choose $m \geq 2(F+1)\log(2e) + 2\log U$. This concludes our proof.

The lower bound is true due to the fact that any set of ω trees with fixed structure has the above VC dimension. The result can be achieved by setting in each tree (beside one) a neutralized terminal node (i.e. a terminal node with posteriori probabilities that are equal to the apriority probabilities).

Our preliminary experiments have shown that estimating the generalization error by using the lower bound of the VC Dimension provides a better performance.

5. Empirical Test

In order to validate the D-IFN algorithm and to illustrate the usefulness of the decomposition approach in classification problems, we apply our method to various domains and compare the results to other methods. In this paper we decided to compare the D-IFN algorithm to the following algorithms: IFN, Naive Bayes and C4.5. We have special interest in both IFN and Naive Bayes as they represent specific points in the search space of the D-IFN algorithm. We also compare the D-IFN to C4.5, because its similarity to IFN (in the model structure) and due to its popularity in many other studies.

The D-IFN approach has been applied to 15 public domain dataset from the UCI Machine learning [16].

Table 1 shows the average accuracy and sample standard deviation obtained by using 10-fold-cross-validation. One tailed paired t-test with confidence level of 95% was used in order to verify whether the differences in accuracy are statistically significant. The superscript "+" or "-" indicates that the accuracy rate of D-IFN was significantly higher or lower than the corresponding algorithm. The results of our experimental study are very encouraging. In fact there was no significant case where Naive Bayes or IFN were more accurate than D-IFN, on the other hand D-IFN was more accurate than Naive Bayes and IFN in 8 databases. and 7 databases respectively. Furthermore D-IFN was significantly more accurate than C4.5 in 8 databases, and less accurate in only 2 databases.

In order to understand when the suggested approach introduces considerable performance enhancement, we checked the correlation between error reduction and the problem complexity.

There are two obvious alternatives for measuring the error reduction achieved by using Attribute Decomposition approach: measuring the error difference between IFN and D-IFN or measuring the error ratio (i.e. the error of D-IFN divided by the error of single IFN). Following [1] we use error ration because it manifests the fact that it becomes gradually harder to achieve error reduction as the error of single IFN converge to zero.

In order to estimate the problem complexity we used the following ratio: the log of the hypothesis space size divided by the training set size.

The estimated linear correlation coefficient (r) is 0.9. This result is quite encouraging as it evidently indicates when should we potentially use attribute decomposition.

Table 1. Summary of experimental results

Database	Naive Bayes	C4.5	IFN	D-IFN
Aust	84.93±2.7	85.36±5.1	84.49±5.1	84.49±2.9
Bcan	97.29±1.6	$^{+}$92.43±3.5	$^{+}$94.39±3.5	97.29±1.6
LED17	$^{+}$63.18±8.7	$^{+}$59.09±6.9	$^{+}$55.55±6.3	73.64±5.5
LETTER	$^{+}$73.29±1	$^{+}$74.96±0.8	$^{+}$69.56±0.7	79.07±0.9
Monks1	$^{+}$73.39±6.7	$^{+}$75.81±8.2	$^{+}$75.00±10.7	92.74±11
Monks2	$^{+}$56.21±6.1	$^{+}$52.07±8.6	63.87 ±6.4	62.13±6.4
Monks3	93.44±3.7	93.44±3.7	92.38±3.3	92.62±3.3
MUSH	$^{+}$95.48±0.9	100±0	100±0	100±0
Nurse	$^{+}$65.39±24	$^{-}$97.65±0.4	92.47±0.5	92.67±0.6
OPTIC	91.73±1.3	$^{+}$62.42±2	$^{+}$48.90±2.5	91.73±1.4
Sonar	75.48±7.3	69.71±5.4	76.48±6.8	77.12±8.7
SPI	$^{+}$94.2±0.9	$^{+}$91.2±1.9	$^{+}$87.00±2.6	95.8±0.9
TTT	$^{+}$69.27±3.2	$^{-}$85.31±2.7	73.19±3.9	73.33±4
Wine	96.63±3.9	$^{+}$85.96±6.9	$^{+}$91.45±5	96.63±3.9
Zoo	89.11±7	93.07±5.8	90.89±9.7	92.71±7.3

6. Summary and Future Work

In this paper we presented a new concept of attribute decomposition for classification problems and proposed the D-IFN algorithm for discovering the appropriate decomposition. The algorithm has been implemented on variety of datasets, and has generated models with higher classification accuracy than other comparable methods.

Finally, the issues to be further studied include: considering other search methods for the problem defined, examining how the attribute decomposition concept can be implemented using other classification methods like Neural Networks, examining other techniques to combine the generated classifiers, and exploring different decomposition paradigms other than attribute decomposition.

Along with performing more empirical, we hope to better understand when it will be appropriate to use attributes decomposition.

References

[1] K. M. Ali and M. J. Pazzani. Error reduction through learning multiple descriptions. *Machine Learning*, 24(3):173–202, 1996.

[2] F. Attneave. *Applications of Information Theory to Psychology*. Holt, Rinehart and Winston, New York, 1959.

[3] W. Buntine. Graphical Models for Discovering Knowledge, in U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pp 59-82. AAAI/MIT Press, 1996.

[4] P. Domingos and M. J. Pazzani. On the the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[5] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*, New-York, NY: Wiley, 1973.

[6] G. Dunteman. *Principal Components Analysis*, Sage Publications, 1989.

[7] J. Elder IV and D. Pregibon. A Statistical Perspective on Knowledge Discovery in Databases, in U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pp 83-113. AAAI/MIT Press, 1996.

[8] U. Fayyad, G. Piatesky-Shapiro, and P. Smyth. From Data Minig to Knowledge Discovery: An Overview, in U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pp 1-30, AAAI/MIT Press, 1996.

[9] J. Friedman and J. Tukey. A Projection Pursuit Algorithm for Exploratory Data Analysis, *IEEE Transactions on Computers*, 23 (9): 881-889, 1974.

[10] J. Kim and C. Mueller. *Factor Analysis: Statistical Methods and Practical Issues*. Sage Publications, 1978.

[11] I. Kononenko. Semi-naive bayesian classifier. *Proceedings of the Sixth European Working Session on Learning*, pages 206–219, 1991.

[12] Liu and Motoda. *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic Publishers, 1998.

[13] O. Maimon and M. Last. *Knowledge Discovery and Data Mining: The Info-Fuzzy network (IFN) methodology*, Kluwer Academic Publishers, 2000.

[14] O. Maimon and L. Rokach. Data Mining by Attribute Decomposition with semiconductors manufacturing case study in D. Bracha, Editor, *Data Mining for Design and Manufacturing: Methods and Applications*, Kluwer Academic Publishers, 2001.

[15] Y. Mansour and D. McAllester. Generalization Bounds for Decision Trees, *COLT 2000*: 220-224.

[16] C. Merz and P. Murphy. UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[17] B. Pfahringer. Controlling constructive induction in CiPF, *Proceedings of the European Conference on Machine Learning*, Springer-Verlag, pp. 242-256. 1994.

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

[19] G. Ridgeway, D. Madigan, T. Richardson, and J. O'Kane. Interpretable Boosted Naive Bayes Classification, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp 101-104, 1998.

[20] M. Schmitt. On the complexity of computing and learning with multiplicative neural networks, to appear in Neural Computation, 2001.

[21] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[22] C. Wallace. MML Inference of Predictive Trees, Graphs and Nets, *Computational Learning and Probabilitic Reasoning*, A. , Gammerman (ed), Wiley, pp 43-66, 1996.

[23] R. Walpole and R. Myers. *Probability and Statistics for Engineers and Scientists*, pp. 268-272, 1986.

[24] B. Zupan, M. Bohanec, J. Demsar, and I. Bratko. Feature transformation by function decomposition, *IEEE intelligent systems & their applications*, 13: 38-43, 1998.