

# Exploiting Hierarchical Domain Values in Classification Learning \*

Yiqiu Han and Wai Lam

Department of Systems Engineering and Engineering Management

The Chinese University of Hong Kong

Shatin

Hong Kong

{yqhan,wlam}@se.cuhk.edu.hk

**keywords:** hierarchical features, heterogeneous features, parameter estimation, classification

## Abstract

We propose a framework which can exploit hierarchical structures of feature domain values to improve classification performance. Mean-variance analysis method under this framework is investigated. One characteristic of our framework is that it provides a principled way to transform an original feature domain value to a coarser granularity by utilizing the underlying hierarchical structure. Through this transformation, a tradeoff between precision and robustness is achieved to improve the parameter estimation in classification learning. We have conducted an experiment using a biological data set and. The results demonstrate that utilizing domain value hierarchies gains benefits for classification.

## 1 Introduction

Hierarchical structures are commonly found in data sets of various applications. Previous machine learning methods mainly focused on utilizing the hierarchy of class labels. The classification task is divided into a sequence of sub-class classification tasks where each subclass is associated with a certain node in the class label hierarchy. Other than class hierarchy, some data sets may contain features with categorical domain values organized in a hierarchical structure. Such a hierarchy reflects existing knowledge about feature values and reveals their inter-relationships in different levels.

In this paper, we refer *hierarchical feature* to a feature with categorical values organized in a hierarchical structure. For example, the attributes handled by “drill down” or “scroll up” operations in Online Analytical Processing (OLAP) are basically hierarchical fea-

tures. Consider a “protein class” attribute in the data set from KDD Cup 2002 Task 2, it may take a value in the form of “GTP-binding proteins/trimeric GTP-proteins/alpha subunits”. “Alpha subunits” is the most specific description while it together with other subunits constitutes a more general concept of “trimeric GTP-proteins”. “trimeric GTP-proteins” and other families constitute “GTP-binding proteins”, and so on. This agglomerative view is based on a hierarchy which consists of different levels of biological concepts. The hierarchical structure of feature domain values can be treated as a tree and each node on the tree corresponds to a distinct value in the domain of the feature. The value represented by a parent node denotes a more general concept with respect to the value represented by a child node. In many data sets, only the leaf nodes represent valid domain values and the non-leaf nodes are treated as “virtual” domain values. In practice, however, it may be difficult and quite expensive to precisely collect the most specific feature value for every data record. The information can be quite incomplete. Hence, the non-leaf nodes can also be valid domain values as well.

The problem of data sparseness arises as the number of valid domain values becomes large for a hierarchical feature. Traditional model parameter estimation for learning becomes very unreliable since a large domain of feature values reduces the effective amount of samples used to estimate those parameters. This situation becomes worse when the data set used for training is already sparse.

The key to solve this problem may lie in the hierarchy itself. As stated above, the hierarchy in fact reflects existing knowledge about feature values, revealing their inter-relationships in different levels. This paper demonstrates that if utilized properly, this background knowledge can be helpful in classification. Although it has already been shown that hierarchical structure of feature domain values is helpful in OLAP, little attention has been paid to this kind of hierarchical features

\*The work described in this paper was substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Nos: CUHK 4187/01E and CUHK 4179/03E) and CUHK Strategic Grant (No: 4410001).

in classification problems. Forman [5] engineered additional hierarchy prevalence features for handling the hierarchical features in the data set of KDD Cup 2002 Task 2.

McCallum et al. [1, 9, 10] applied shrinkage [7, 12] technique in a hierarchy of classes and showed some improvements on text classification. Freitag and McCallum [6] adopted the same shrinkage technique in information extraction problems. Segal and Koller [11] introduced a general probabilistic framework for clustering biological data into a hierarchy. Dumais and Chen [4] utilized the hierarchy of classes for web content classification. However, the above methods only consider class hierarchies and do not handle hierarchical feature domain values.

This paper introduces a framework which can exploit the hierarchical structure of domain values in classification learning. This framework provides a principled way for transforming an original domain value to a “coarser” one with respect to the domain value hierarchy. Through this transformation, a tradeoff between precision and robustness is achieved to improve the parameter estimation in classification learning. Another characteristic of our framework is that the relationship between the domain value and class distribution is considered by Mean-variance analysis method. Our framework can also be integrated into many existing learning models easily.

We have conducted an experiment using a yeast gene data set from KDD Cup 2002 Task 2. The results show that our proposed framework can improve the classification performance under different learning models, especially in sparse data situations.

## 2 Features with Hierarchical Domain Values

**2.1 Background** In many occasions, the relationship between two discrete domain values can be more than just equality testing. Some domain values may be close to each other in terms of the similarity of the concepts. It is common that the domain values of a particular feature in real world are organized in a hierarchy. One example is the “protein class” attribute mentioned in the gene data set of KDD Cup 2002 Task2. Each node on the tree corresponds to a distinct value or a semantic concept. The value represented by a node at a higher level denotes a more general concept with respect to the value represented by a node in the lower level. From the root to a leaf, the concept becomes more and more specific.

**2.2 Parameter Estimation for Learning** To illustrate how to exploit hierarchical features in classification learning, we consider parameter estimation in classifica-

tion learning. Many classification models make use of a probability distribution defined over a set of parameters, denoted as  $\theta$ . Let  $C = \{c_1, \dots, c_{|C|}\}$  denote the set of classes. The training data is used to estimate some model parameters. After that, the model equipped with the probabilistic parameters can be used for predicting the class label or calculating the membership likelihood for a certain class  $c_j$ .

Suppose  $F = (f_1, \dots, f_m)$  denotes the set of feature values of a new instance to be classified. There are different parametric models for classification. The estimation of  $P(C|F)$  is usually the key for classification. For example, adopting the commonly used conditional independence assumption of features [3], the estimation can be conducted by the following formula:

$$L(C|F = (f_1, \dots, f_m)) = \prod_{i=1}^m L(C|F_i = f_i) = \prod_{i=1}^m \frac{\Theta(C, F_i = f_i)}{\Theta(F_i = f_i)} \quad (2.1)$$

where  $L(C|F = (f_1, \dots, f_m))$  is the likelihood score for the estimation of  $P(C|F)$ ;  $L(C|F_i = f_i)$  is the conditional likelihood score for class  $C$  given that the value of  $F_i$  is  $f_i$ ;  $\Theta(F_i = f_i)$  is the number of training instances with  $F_i$  instantiated to the value  $f_i$ ; and  $\Theta(C, F_i = f_i)$  is the number of training instances with class label  $C$  and  $F_i$  instantiated to  $f_i$ .

Take the classical Naive Bayesian model for example, classification can be expressed as a parametric model  $P(C|F) = P(C)P(F|C)/P(F)$ . One can derive from this expression and show that  $L(C|F_i)$  acts as an important model parameter. Given a set of training instances, one common estimation technique is the maximum likelihood estimation, which is usually based on the familiar ratios of empirical counts. It can be observed that when the domain space is large and the amount of positive instances for a class is small, parameters such as  $L(C|F_i)$  cannot be estimated reliably.

## 3 Exploiting Domain Value Hierarchy

### 3.1 Expanding Hierarchical Domain Values

The score  $L(C|F_i)$  as given in Equation 2.1 for a hierarchical feature  $F_i$  can be achieved by computing  $\Theta(C, F_i)/\Theta(F_i)$ , which is the ratio of empirical counts. First we can collect all the training instances with a specific value for this particular feature  $F_i$ . Then  $\Theta(C, F_i)/\Theta(F_i)$  can be calculated as the ratio of the instances with class  $C$  in the collection.

Let  $L(C|F_i)$  be the observation for estimating  $P(C|F_i)$ . The training instances collected for  $F_i$  can be regarded as  $\Theta(F_i)$  i.i.d. Bernoulli trials with success probability  $P(C|F_i)$ . Therefore, the observed value of  $L(C|F_i)$  has the following expectation and variance:

$$E(L(C|F_i)) = P(C|F_i) \quad (3.2)$$

$$\text{Var}(L(C|F_i)) = \frac{P(C|F_i)(1 - P(C|F_i))}{\Theta(F_i)} \quad (3.3)$$

According to Chebyshev theorem:

$$P(|L(C|F_i) - P(C|F_i)| \geq \epsilon) \leq \frac{P(C|F_i)(1 - P(C|F_i))}{\Theta(F_i)\epsilon^2} \quad (3.4)$$

where  $\epsilon$  is a small constant that denotes an arbitrarily small error bound.

Equation 3.4 shows that if we use  $L(C|F_i)$  to estimate  $P(C|F_i)$ , the variance increases while the value of  $\Theta(F_i)$ , representing the amount of training instances with a specific  $F_i$  value, is decreasing. In cases where the size of training data set is small or the number of domain values for a hierarchical feature is large, any observed  $\Theta(F_i)$  and  $P(C|F_i)$  will tend to be very small. Hence the variance of  $L(C|F_i)$  may exceed  $1/\Theta(F_i)$  of  $P(C|F_i)$ . Under this circumstance, traditional learning models which require parameter estimations become very unreliable and thus affecting the classification performance.

To cope with the problem, we propose a framework to exploit the hierarchical structure of feature domain values. The main idea is inspired by a statistical technique called *shrinkage* which provides a way to improve parameter estimation so that it can deal with a limited amount of training data [7, 12]. Since the problem is caused by insufficient  $\Theta(F_i)$ , the idea is to replace  $L(C|F_i)$  with another maximum likelihood estimation with respect to a coarse granularity of the specific domain value, denoted by  $T(F_i)$ .  $T(F_i)$  is defined according to both the value of  $F_i$  and the domain value hierarchy. Suppose the feature  $F_i$  of an instance is instantiated with a certain value  $f_i$ , then this value is replaced by a new value corresponding to  $T(F_i = f_i)$ , but not vice versa. The domain value corresponding to  $T(F_i = f_i)$  encompasses the original domain value  $f_i$  together with an expanded boundary in the domain value space. Specifically, the domain value for  $T(F_i = f_i)$  consists of a set of domain values including  $f_i$ .

Intuitively,  $T(F_i = f_i)$  is associated with a larger amount of training instances, which are expected to have similar characteristics as the collection for  $f_i$ . This derived domain value,  $T(F_i = f_i)$ , has its own  $\Theta(T(F_i = f_i))$  and  $\Theta(C, T(F_i = f_i))$ , both defined in a similar way as in Equation 2.1. Then a modified maximum likelihood score  $L(C|T(F_i = f_i))$  can be defined as follows:

$$L(C|T(F_i = f_i)) = \frac{\Theta(C, T(F_i = f_i))}{\Theta(T(F_i = f_i))} \quad (3.5)$$

This process attempts to maintain a tradeoff between the precision and robustness of parameter estimation in classification learning.

The next issue is how to obtain the  $T(F_i) = f_i$  given  $F_i = f_i$ . The topological parent-child link between two nodes usually suggests a generalization/specialization relationship between corresponding domain values. The knowledge underlying this kind of relationship has been shown to be helpful in learning under a hierarchy of classes by McCallum et al. [9]. An improved estimate for each leaf node is derived by “shrinking” its maximum likelihood estimate towards a linear interpolation of the maximum likelihood estimates of its ancestors. In our framework, when handling domain value hierarchies, we consider those sub-trees enclosing the target node  $F_i$  as possible candidates for  $T(F_i)$ . This processing of domain values can be regarded as a kind of expansion.

If the scope of the sub-tree  $T(F_i)$  is properly selected, it will provide a collection of related training instances that can produce a more robust estimation for  $P(C|F_i)$ . But if the scope is too narrow, it cannot improve the robustness of the parameter estimation. On the other hand, if the scope is too large, the estimation will become imprecise. We propose a principled method to obtain an appropriate  $T(F_i)$ .

**3.2 Mean-Variance Expansion** The second proposed method is based on mean-variance analysis, called MVE (Mean-Variance Expansion) algorithm. The expansion is conducted as long as the upper bound of the expected parameter estimation error does not increase. Since  $T(F_i)$  is defined as a refined  $F_i$ , we assume that the posterior probability  $P(C|F_i)$  follows a distribution with mean  $P(C|T(F_i))$  and variance  $\sigma_{T(F_i)}^2$ . Then, the upper bound of the error in using the posterior probability of  $T(F_i)$  to estimate that of  $F_i$  is given as follows:

$$P(|P(C|F_i) - P(C|T(F_i))| \geq \epsilon) \leq \frac{\sigma_{T(F_i)}^2}{\epsilon^2} \quad (3.6)$$

where  $\epsilon$  is a small constant denoting an arbitrarily small error bound.  $\sigma_{T(F_i)}^2$  is independent of the variance of  $L(C|F_i)$  or  $L(C|T(F_i))$  and can only be determined by the position of  $T(F_i)$  in the domain value hierarchy as well as the way in which  $T(F_i)$  is partitioned into different branches.

Let  $\psi$  be the error of using the observed posterior probability  $L(C|T(F_i))$  to estimate  $P(C|F_i)$  and it is defined as:

$$\begin{aligned} \psi &= L(C|T(F_i)) - P(C|F_i) \\ &= (L(C|T(F_i)) - P(C|T(F_i))) + \\ &\quad (P(C|T(F_i)) - P(C|F_i)) \end{aligned} \quad (3.7)$$

Consider the expectation and variance of  $\psi$

$$E(\psi) = 0 \tag{3.8}$$

$$\text{Var}(\psi) = \sigma_{T(F_i)}^2 + \frac{P(C|T(F_i))(1 - P(C|T(F_i)))}{\Theta(T(F_i))} \tag{3.9}$$

According to Chebyshev theorem,

$$P(|L(C|T(F_i)) - P(C|F_i)| \geq \epsilon) \leq \frac{\sigma_{T(F_i)}^2}{\epsilon^2} + \frac{P(C|T(F_i))(1 - P(C|T(F_i)))}{\Theta(T(F_i))\epsilon^2} \tag{3.10}$$

where  $\epsilon$  is a small constant denoting an arbitrarily small error bound.

Consider both Equations 3.4 and 3.10,  $L(C|T(F_i))$  should be chosen to estimate  $P(C|F_i)$  rather than  $L(C|F_i)$  if the following equation holds:

$$\sigma_{T(F_i)}^2 + \frac{P(C|T(F_i))(1 - P(C|T(F_i)))}{\Theta(T(F_i))} < \frac{P(C|F_i)(1 - P(C|F_i))}{\Theta(F_i)} \tag{3.11}$$

We can iteratively use the observed posterior probability of “ancestor” sub-trees to estimate the original one until Equation 3.11 does not hold. Figure 3.2 depicts the pseudo-code of MVE algorithm. The algorithm searches the appropriate sub-tree  $T(F_i)$  by checking the expected mean and variance of the probabilistic parameters distributed over the domain value hierarchy.

---

```

// Given the ith hierarchical feature
// Estimate the  $\sigma_{T(F_i)}^2$ 
FOR j = 1 to n
  LET  $F_i = f_{ij}$ 
  IF  $F_i$  corresponds to a leaf node
     $\sigma_{F_i}^2 = 0$ 
  ELSE
     $\hat{\sigma}_{T(F_i)}^2 = \text{Avg}((L(C|T(F_i)) - L(C|F_i))^2)$ 
// Expand the domain values;
For j = 1 to n
  IF  $f_{ij}$  not expanded
     $F_i = f_{ij}$ 
     $T(F_i) = F_i$ 
  LOOP
    // Parent( $T(F_i)$ ) returns the parent sub-tree
    // of the sub-tree associated with  $T(F_i)$ .
     $T(F_i) = \text{Parent}(T(F_i))$ 
  UNTIL Equation 3.11 does not hold
  Return  $T(F_i)$  as a new nominal value.

```

---

Figure 1: The pseudo-code of Mean-Variance Expansion algorithm (MVE)

## 4 Experimental Results

The first data set used in our experiment is a yeast gene data set obtained from KDD Cup 2002 Task 2 [2]. Every instance in the data set corresponds to a gene (gene-coded protein) and the class label is based on the discrete measurement of how active one (hidden) system is when the gene is knocked out (disabled). Each instance is represented by a set of features including three hierarchical ones, namely, *localization*, *protein class*, and *functional class*.

The data set is very sparse and noisy, reflecting existing knowledge of the yeast genes. There are plenty of missing feature values, aliases, and typographical errors. There are 6,400 instances. 3,018 instances are used for training and 1,489 instances are held aside for testing. Only 84 genes in the training set are related to the hidden system. 38 of these genes are labeled with “change” that interests biologists. The other instances in those 84 genes are labeled with “control”, meaning that they can affect the hidden system indirectly. The remaining 2,934 genes are labeled with “nc”, meaning that they do not affect the hidden system.

The objective is to return a ranked list of genes in the testing set in the order of their likelihood of belonging to an interested category. The first category is called “narrow” classification whose aim is to predict the class “change”. The second category is called “broad” classification whose aim is to predict the union of the classes “change” and “control”.

The evaluation metric for the classification performance on this yeast gene data set is based on a score metric offered by the KDD Cup 2002 Task 2. This metric is called Receiver Operating Characteristic (ROC) score. An ROC curve is a plot of the true positive rate (TP rate) against the false positive rate (FP rate) for different possible thresholds. Under a particular threshold, TP rate is the fraction of the positive instances for which the system predicts “positive”. The FP rate is the fraction of the negative instances for which the system erroneously predicts “positive”. The area under the plot is the ROC score.

Figure 2 depicts the classification performance of Naive Bayesian and our ENE method integrated with Naive Bayesian for the “narrow” classification problem. We plot the performance of our three methods and Naive Bayesian model. The diagonal straight line represents random prediction. The ROC curve of Naive Bayesian is similar to the random prediction. It shows that the Naive Bayesian learning cannot handle the data sparseness problem in this gene data set. Meanwhile, our method shows significant improvement on this gene data set. The ROC curve is not smooth because the gene data has only less than twenty positive instances

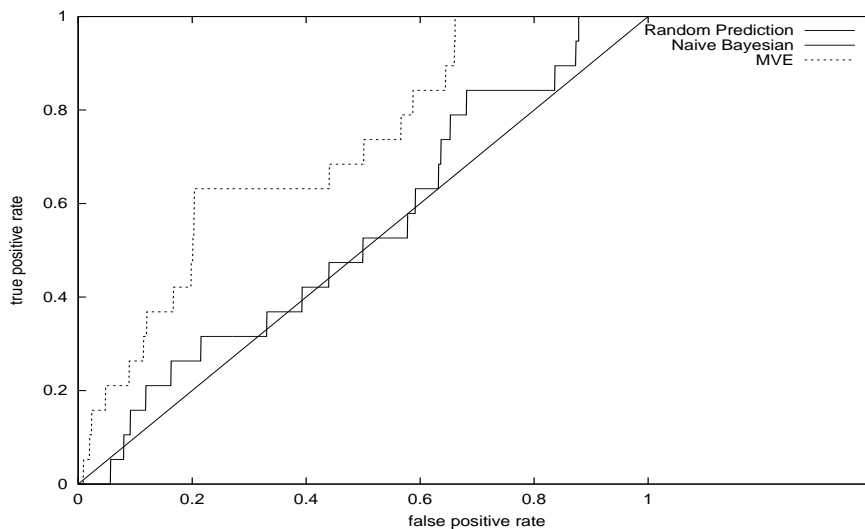


Figure 2: Classification performance measured by ROC curve on the gene data set

out of thousands of testing instances.

Tables 1 shows the classification performance measured by ROC score for the “broad” classification problem. It shows our proposed method obtains significant improvements on the Naive Bayesian learning model. It demonstrates that our framework can handle learning from extremely sparse data and offer a more reliable result. As for reference, Kowalczyk and Raskutti [8] reported their solution with the winning ROC score in the KDD Cup 2002 Task 2. The winning score for the “broad” classification task was 0.684. The score distribution of submitted results is presented in [2].

	Naive Bayesian	MVE
Broad	0.512	0.694
Narrow	0.520	0.713

Table 1: Classification performance measured by ROC score on the gene data set using the same training/testing splitting as in KDD Cup 2002 Task 2

## References

- [1] L. D. Baker and A. K. McCallum, *Distributional Clustering of Words for Text Classification*, Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 96–102, 1998.
- [2] M. Craven, *The genomics of a signaling pathway: A KDD cup challenge task*, SIGKDD Explorations, 4(2):97–98, 2003.
- [3] P. Domingos and M. Pazzani, *Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier*, Machine Learning 29, pp. 103–130, 1997.
- [4] S. Dumis and H. Chen, *Hierarchical Classification of Web Content*, Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 256–263, 2000.
- [5] G. Forman, *Feature Engineering for a Gene Regulation Prediction Task*, SIGKDD Explorations, 4(2):106–107, 2003.
- [6] D. Freitag and A. McCallum, *Information Extraction with HMMs and Shrinkage*, Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction, 1999.
- [7] W. James and C. Stein, *Estimation with Quadratic Loss*, Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability 1, pp. 361–379, 1961.
- [8] A. Kowalczyk and B. Raskutti, *One class SVM for yeast regulation prediction*, SIGKDD Explorations, 4(2):99–100, 2003.
- [9] A. McCallum, R. Rosenfeld, T. Mitchell and A. Y. Ng, *Improving Text Classification by Shrinkage in a Hierarchy of Classes*, Proceedings of the International Conference on Machine Learning (ICML), pp. 359–367, 1998.
- [10] A. McCallum and K. Nigam, *Text classification by bootstrapping with keywords, EM and shrinkage*, ACL Workshop for Unsupervised Learning in Natural Language Processing, 1999.
- [11] E. Segal and D. Koller, *Probabilistic Hierarchical Clustering for Biological Data*, Annual Conference on Research in Computational Molecular Biology, pp. 273–280, 2002.
- [12] C. Stein, *Inadmissibility of the Usual Estimator for the Mean of a Multivariate Normal Distribution*, Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability 1, pp. 197–206, 1955.