Mean-entropy Discretized Features are Effective for Classifying High-dimensional Bio-medical Data

Jinyan Li[*]
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
jinyan@i2r.a-star.edu.sg

Huiqing Liu
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
huiqing@i2r.a-star.edu.sg

Limsoon Wong
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
limsoon@i2r.a-star.edu.sg

ABSTRACT

This paper studies an empirical feature selection heuristics for classifying high-dimensional bio-medical data. A feature's discriminating power can be measured by its entropy value. Based on this idea, we do not consider those features that are ignored by the entropy idea. Such a selection can usually reduce the dimensionality of the data by 90–95%. Then we rank the remaining features, and select features whose entropy is smaller than the average of all the remaining features' entropies. This round of selection can usually further reduce two thirds of the features. So, we can achieve a reduction from tens of thousands of features to only hundreds of important features. Furthermore, we also observe that learning algorithms, including our new tree-committee classifier, generally improve their accuracy after the feature selection. This heuristics appears to be more systematic than the prevailing use of specific numbers of top-ranked features for classification.

General Terms

Bioinformatics

Keywords

Gene expression data, Proteomic profiling data, Feature selection

1. INTRODUCTION

Many bio-medical applications are supervised learning problems. Clinical diagnosis is to classify whether a sample is normal or abnormal [1]; prognosis is to predict, at the time of diagnosis, whether a patient will relapse or not after treating with an existing standard therapy [18]; subtype distinction is to identify correct sub-classes of patients who suffer from one heterogeneous disease [18]. With the advances in wet-experimental technologies, the data become ever larger, and a more challenging part of the data is the large number of features or dimensions. For example, the microarray gene expression profiling technology can simultaneously measure the expression levels of tens of thousands of genes, translating to a relational data set that is described by tens of thousands of features.

Feature selection is therefore crucial for analysing high dimensional bio-medical data. Reasons include: (1) It is impossible for biologists or doctors to examine the whole feature space (e.g. the genes in human genome) by laboratory experiments at one time. A small percentage of the features should be recommended by computational algorithms to be focused on in the laboratory experiments. Then, the concentration on these recommended features may help experts to understand better and deeper about some biomedical mechanism. (2) Many features are irrelevant to the classification. Taking such features into account during classification increases the dimensionality of the problem, raises many computational difficulties, and potentially introduces noise effect on the classification accuracy [2; 12]. So, how to select important features for classification is a problem that has been attracting tremendous research effort previously and currently.

In this paper, we suggest a method for feature selection and for rule discovery. This method differs from the use of 'user-favorate' numbers of top-ranked features in the analysis on high-dimensional gene expression profiling data and proteomic mass/charge profiling data. Those numbers were very specific—for example, 50, 100, 200, or 1000, and so on—different from one research to another [1; 11; 18].

Our idea is to use entropy-discretized features whose entropy value is *smaller* than the *mean* entropy value of all the entropy-discretized features. Note that we define entropy-discretized features as features that are discretized into at least two intervals by an entropy-discretization method [9]. In our method, we do not consider features that are ignored by the entropy method. Our selected features are called *mean-entropy discretized* features. According to our experience, this idea can filter out 90% - 95% of the original features for different data sets, significantly reducing the dimensionality of the problems. This idea can also pinpoint ideally discriminating features— those features can be individually used to make 100% clear distinction between classes.

We present experimental results to show that classification accuracy can be constantly improved, sometimes significantly,

[*]Correspondence author.

if these entropy discretized features are used in classification algorithms, instead of the whole feature space. This is a significant result because we use much less features but we get better accuracy. We also compare our accuracy results with those when specific numbers (20, 50, 100, and 200 are studied in this paper) of top-ranked features are considered for classification. We found that the accuracy of a learning algorithm on different data sets fluctuated in contrasting curve shapes when different numbers of top-ranked features are used. So, it is hard to observe a regular guideline for selecting a fixed number of top-ranked features for best results.

An important purpose in the analysis of biomedical data is to discover interactions or causal relationships among features. This paper proposes a rule-based classifier that can capture this kind of knowledge patterns from data. This classifier is a new ensemble method [15], named CS4, consisting of a committee of *cascading* decision trees. Each tree is constructed by using one of the top-ranked features as its root node. So, in a cascading manner, we can build k trees by using top k features as root node. In high-dimensional data sets, a small k number (e.g. 20) of top-ranked features usually have almost the same merit such as similar entropy values or gain ratios. So, it is reasonable and fair to use different top-ranked features as root node. The cascading trees are voted in a weighted manner to make a final decision when a test sample is presented. CS4 differs from the traditional Bagging [4] and Boosting [10] ideas because they use bootstrapped data in the construction of committee trees. However, we always use the same original training data throughout the construction of the committee. We use experimental results to show that the performance of CS4 is better than the Bagging and Boosting algorithms.

The remainder of the paper is organized as follows: Section 2 presents our feature selection ideas and reviews a core discretization algorithm [9] that is used in our method for the first round of selection. Section 3 describes our a newly proposed committee classifier consisting of k cascading decision trees. Section 4 outlines five state-of-the-art classifiers and 15 high-dimensional biomedical data sets. Section 5 reports our experimental results to show that our feature selection method is effective for improving the classifiers' accuracy. In addition, we also discuss the stability, comprehensibility, and wrapped features of the classifiers.

2. USING MEAN ENTROPY VALUES AS SPLITTER FOR FEATURE SELECTION

This section presents our idea and steps for feature selection. The main algorithm is the entropy discretization method [9], which is also reviewed in this section.

2.1 Our Feature Selection Idea

Our idea is to first rank all individual features according to their entropy value, then we use the average entropy value as a splitter to cut off all the lower ranked features. The steps are as follows:

1. Rank all features into an ascending order according their entropy values [9],
2. Remove those features that are ignored (not discretized) by the entropy-discretization method,
3. Calculate the average of the entropy values of the remaining features, and

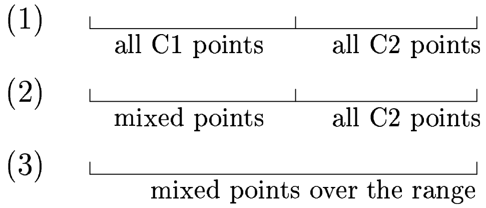


Figure 1: Distributions of a range of points with two class labels.

4. Select those features that have smaller entropy values than the average for classification.

Note that features with a smaller entropy are more discriminating. The feature reduction rate on 5 high-dimensional data sets is shown in Table 1. (The description of the data sets is given in Section 4.) Usually, our idea can reduce about 90%-95% of the dimensionality of the problems in the first round of selection as per Step 2, and reduce about two thirds of the remaining features in the second round of selection as per Step 4.

2.2 The Entropy Discretization Method

We first explain the basic idea of the entropy-based discretization method [9]. For a range of real values in which every point is associated with one of two class labels, the distribution of the labels can have three main basic shapes as shown in Figure 1: (1) Big intervals each containing the same class of points; (2) Big intervals but not all of them containing a same class of points; (3) Class points randomly mixed over the range. Using the middle point between the two classes, the entropy method partitions the range in the case of Figure 1(1) into two intervals. The entropy of such a partitioning is 0. For the case of Figure 1(2), the method partitions the range in such a way that the right interval contains as many C2 points as possible and contains as few C1 points as possible. This is to minimize the entropy of this feature. For the case of Figure 1(3), the method ignores the feature as mixed points over a range do not provide rules for reliable classification. That a range is partitioned into at least two intervals is called *discretization*. In general, ideally discriminating features (as shown in Figure 1(1)), sub-optimal features (as shown in Figure 1(2)), and those features with random class distributions can be effectively identified by the entrop-based discretization method.

Next we present an example [14] of ideally discriminating features discovered by the entropy method from a gene expression profiles. The data is to differentiate one subtype (E2A-PBX1) from the other subtypes of the heterogenious childhood leukemia disease [18]. The training data consist of 18 E2A-PBX1 cells and 197 cells of other subtypes; the test data consist of 9 E2A-PBX1 cells and 103 cells of other subtypes. This data set have 12558 features each describing the expression range of a gene.

For example, gene 32063_*at* is an ideally discriminating features discovered from the above training data. The *cut point* for this feature is 4068.7, which partitions the expression range of this gene into two intervals, $[0, 4068.7)$ and $[4068.7, +\infty)$. Note that this gene's expression of all E2A-PBX1 samples were ≥ 4068.7, falling into the right interval; however its expression of any other subtype samples was

Table 1: Feature reduction rates on high-dimensional bio-data sets.

Data sets	Number of features			Reduction rates
(Training data)	original	entropy discretized	mean-entropy discretized	1st, 2nd round
T-ALL	12558	1309	415	89.6%, 68.3%
E2A-PBX1	12558	718	235	94.3%, 67.3%
BCR-ABL	12558	84	31	99.3%, 63.1%
ALL-AML	7129	866	350	87.9%, 59.6.%
Lung cancer	12533	2173	777	82.7%, 64.2%

less than 4068.7, falling into the left interval. The entropy of this partitioning for this feature is the minimal value of 0. This cut point also clearly separates the reserved 112 test samples with no mistakes.

The discretization method also discovered 713 sub-optimal features from this data set, and ignored about 11840 features (94.28% of the whole feature space) that had a random expression distribution without any interval covering a sufficient percentage of the two classes of samples. Obviously, these features are irrelevant for classification.

Using our mean-entropy idea as discussed earlier, we can further remove 483 features from the 713 sub-optimal features. Overall, we reduced a total of 12558 features to only 235 features.

A formal description of the discretization method can be found in [8; 9].

3. OUR CS4 CLASSIFIER

The widely-used C4.5 decision trees [17] can derive rules and feature interactions from data. Another advantage of tree-based classifiers is: The features involved (wrapped) in a tree is far less than the number of the features describing the training data. For non-linear classifiers such as support vector machines or k-nearest neighbours, the entire feature space must be used in the learning models. However, single C4.5 decision trees have been found to be difficult to often maintain a good performance on test data when handling high-dimensional bio-medical data [15]. In this paper, we introduce a new ensemble classifier of decision trees, called CS4 [15], that can derive many decision trees, and thus many true and significant rules from training data, and that has very good test accuracy.

The learning phase of the CS4 classifier is to construct a certain number of trees. Suppose n number of features describe a given data. To construct k ($k \leq n$) number of trees, we use the following steps:

Step 1: Use gain ratios to rank all the features into an ordered list with the best feature at the first position.

Step 2: $i = 1$.

Step 3: Use the ith feature as root note to construct the ith tree.

Step 4: Increase i by 1 and goto Step 3, until $i = k$.

Usually we set the number of trees k as 20. Note that there are no changes to the original training data throughout the k iterations. So, all our rules are true when applied to the training data. This is an advantage of our method over Bagging and Boosting for their rules are not always true. Steps 1 and 3 reflect our cascading idea: the root node of the trees shifts from the most important feature to the kth most important feature. We use C4.5 [17] to build the first tree; the construction of the remaining trees uses a method that is slightly different from C4.5: the root node selection is forced, but all the other nodes are selected normally as in traditional C4.5. The number 20 is a heuristic choice. It may be controlled by the gain ratio trend of top-ranked features. This will be one of our future research topics.

To share the discriminating power of the k trees, we propose to use our previous PCL idea [16; 14] to summarize the individual decisions. This sharing idea differs from the simple equal voting approach as adopted by Bagging [4].

We examine the accuracy change trend of this new classifier when the whole feature space, the all entropy-discretized features, or the mean-entropy discretized features are applied. We also examine whether CS4 is resistant to feature selection—whether it can keep good and stable accuracies with little variance after feature selections.

4. OTHER CLASSIFIERS AND DATA SETS

We also examine the accuracy change trend of the state-of-art classifiers such as support vector machines (SVM), Naive Bayes (NB), k-nearest neighbours (k-NN), and C4.5 (Bagging and Boosting). This is aimed to get a fuller picture about the effect of our feature selection method on different classification methods.

SVMs [5; 6] are a kind of blend of linear modeling and instance-based learning. A SVM selects a small number of critical boundary samples from each class and builds a linear discriminant function that separates them as widely as possible. In the case that no linear separation is possible, the technique of "kernel" is used to automatically inject the training samples into a higher-dimensional space, and to learn a separator in that space. The SVM used in this paper is a version that uses polynomial kernels. The k-NN classifier [7] is a long-studied instance-based prediction model. By k-NN, the class label of a test sample is decided by the majority class of its k closest neighbors based on their Euclidean distance. In our experiments, k is set as 3. Naive Bayes [13] is a probabilistic learner based on Bayes's rule. It is among the most practical approaches to certain types of learning problems. Bagging [4] and Boosting [10], the two most widely used ensemble approaches, can improve the performance of a single base classifier. In this paper, we use C4.5 [17] as the base classifier.

The softwares used in this paper is *Weka* version 3.2. Its Java-written open source codes are available at `http://www.cs.waikato.ac.nz/~ml/weka/` under the GNU General Public Licence. Note that we revised a base Java class to discretize a feature: Instead of using boundary points as cutting points, we use the middle points of two classes' boundary points. In implementing our new algorithm CS4, we called some classes in weka.classifiers and also some other APIs. For the committee classifiers—Bagging, Boosting, and our CS4 classifier, we set the number of base classifiers as 20.

The data sets used in this study all come from our Kent Ridge Biomedical Data Sets Repository at `http://sdmc.lit.org.sg/GEDatasets/Datasets.html`.

Most of the data sets are described by more than 10,000 features. This characteristics is in contrast to the widely-used data sets stored at the UCI Machine Learning Repository [3], where many of the data sets are described by less than 20 features. Table 2 gives the basic information of the data sets.

5. RESULTS REPORT

We report our experimental results in three aspects. Firstly, we report error numbers of six classifiers (SVM, NB, k-NN, CS4, C4.5 Bagging, and C4.5 Boosting) on the 15 data sets when the whole feature space, all entropy-discretized features, and all mean-entropy discretized features are used. We define an error number as the number of samples that are wrongly classified by a classifier. Secondly, we report error numbers of four classifiers on 4 data sets when four specific numbers (20, 50, 100, and 200) of top-ranked features are used for classification. Thirdly, we study the stability of the classifiers—to test whether they are resistant to feature selection; we also study the number of wrapped features in a learning model and the comprehensibility of the learning models.

5.1 The Trend of the Error Numbers of the Six Classifiers

Table 3 reports the classification errors on the 15 data sets of SVM, NB, and k-NN when the three scenarios of features—all the original features, all the entropy-discretized features, and only the mean-entropy discretized features—are used in the 10-fold cross-validation (except the last 4 data sets in Table 3). We can see that:

- SVM made much less numbers of mistakes after the feature selections (either in the first round or after the second round) on the data sets BCR-ABL, MLL, Subtype lynphoma, Stjude testing, and ALL-AML testing. On the other data sets, SVM maintained its performance. From the first round to the second round of feature selection, SVM often decreased or maintained the errors.

- NB almost constantly improved its performance very much on most of the data sets. In general, the trend of the error numbers goes to a lower level when less numbers of features are used.

- Similar to NB, k-NN also improved its performance very much on most of the data sets. In general, the trend of the error numbers goes to a lower level when less numbers of features are used.

Table 4 reports the error change trend of three rule-based committee classifiers—our CS4 classifier, Bagging(C4.5) and Boosting(C4.5). We can see that:

- The three committee classifiers did not change as much in their performance, before and after feature selection. Nonetheless, there was still a slight decrease in total errors after the feature selections (See the last row of Table 4). Such a trend is different from the changes occurred in SVM, NB, and k-NN where the total errors are significantly reduced after the features selections (See the last row of Table 3). This interesting phenomenon is an issue about the stability of classifiers in relation to feature selections, and also an issue about wrapped features in a classifier. We discuss these two points later with more details to see why a classifier is more resistant to feature selection, and why another classifier is more sensitive.

- Compared to the classical Bagging and Boosting, our newly proposed CS4 committee classifier outperformed them with only one exception case on the Hyperdip>50 data set where Boosting made 14 errors, but CS4 made 15 errors.

From both Table 3 and Table 4, we can see that the feature selections have constantly helped the classifiers to improve their performance though only 2–10% of the original features are used. From the results achieved by the first round of feature selection and the results achieved after the second round of selection, we did not see much difference in the error numbers. So, we suggest to select mean-entropy discretized features for classification as the second round of selection can usually reduce two thirds of the features selected in the first round.

Next, we discuss which classifier wins the best performance.

- If using the whole feature space without any selection, CS4, SVM, Bagging, k-NN, Boosting, and NB respectively made 85, 99, 119, 178, 185, and 395 total errors on the 15 data sets;

- If using all the entropy discretized features, SVM, CS4, k-NN, Bagging, NB, and Boosting respectively made 51, 75, 85, 113, 115, and 174 total errors;

- If using only the mean-entropy discretized features, SVM, CS4, k-NN, NB, Bagging and Boosting respectively made 47, 75, 77, 98, 113, and 162 total errors on the 15 data sets.

So, SVM and our CS4 classifier are the best two classifiers irrespective of whether there is feature selection or not.

5.2 When Specific Numbers of Features are Used

Our next set of experiments are aimed to see the error number trend of the classifiers when specific numbers of top-ranked features are changed from a small number to a bigger number. In this paper, we study top 20, 50, 100, and 200 features. Figure 2 depicts the change curves of the performance of SVM, C4.5, k-NN, and CS4 when the different numbers of top-ranked features are used on four data sets (ALL-AML, BCR-ABL, MLL, and Hyperdip>50).

Let's focus our discuss on the performance of SVM first (See Figure 2(1)). The error curve on the Hyperdip>50 data set goes stable when the feature number increases from 20 to 50 and then to 100, but the curve goes up when the

Table 2: Data set description. The total number of samples for each of the first 11 data sets are shown in the 3rd column and for 10-fold cross-validation. For the remaining 4 data sets, only test samples are shown here.

Data sets	# classes	# samples	# features	Usage
T-ALL	2	327	12558	Subtype distinction of leukemia
E2A-PBX1	2	327	12558	Subtype distinction of leukemia
TEL-AML1	2	327	12558	Subtype distinction of leukemia
BCR-ABL	2	327	12558	Subtype distinction of leukemia
MLL	2	327	12558	Subtype distinction of leukemia
Hyperdip>50	2	327	12558	Subtype distinction of leukemia
Ovarian cancer	2	253	15154	Ovarian disease diagnosis
Prostate cancer	2	102	12600	Prostate disease diagnosis
Colon tumor	2	62	2000	Colon tumor diagnosis
ALL-AML	2	72	7129	Two subtypes classification of Leukemia
Subtype lymphoma	2	47	4026	Subtypes classification of lymphoma
Stjude testing	2	112	12558	Six subtypes classification of Leukemia
Lung cancer testing	2	149	12533	Lung cancer diagnosis
ALL-AML testing	2	34	7129	Two subtypes classification of Leukemia
Armstrong testing	3	15	12582	Three subtypes classification of Leukemia

Table 3: The change trend of the error numbers of SVM, NB, and k-NN after the feature selections. Here, "all" represents a classifier considered all the features; "entropy" represents the entropy discretized features; and "M-entropy" represents the mean-entropy discretized features.

Data sets	SVM			NaiveBayes			k-NN		
	all	entropy	M-entropy	all	entropy	M-entropy	all	entropy	M-entropy
T-ALL	1	0	0	29	0	0	8	3	0
E2A-PBX1	1	1	1	25	0	0	1	1	1
TEL-AML1	4	3	4	62	6	4	14	4	4
BCR-ABL	12	8	8	15	20	14	15	9	10
MLL	7	5	2	19	5	3	9	5	4
Hyperdip>50	11	9	11	57	15	17	21	13	16
Ovarian	0	0	0	19	16	14	15	11	10
Prostate	7	8	6	40	15	8	18	10	8
Colon Tumor	11	9	8	25	18	14	19	9	11
ALL-AML	1	2	2	2	0	2	10	2	1
Subtype lymphoma	6	3	2	10	3	3	13	5	5
Stjude testing	32	1	2	82	14	16	20	5	2
Lung cancer	1	1	0	7	2	2	3	1	1
ALL-AML testing	5	1	1	3	1	1	10	6	2
Armstrong	0	0	0	0	0	0	2	2	2
Total Errors	99	51	47	395	115	98	178	85	77

Table 4: The change trend of the error numbers of the three rule-based committee classifiers.

Data sets	CS4			Bagging			Boosting		
	all	entropy	M-entropy	all	entropy	M-entropy	all	entropy	M-entropy
T-ALL	1	1	1	1	1	1	1	1	1
E2A-PBX1	1	1	1	1	1	1	1	1	1
TEL-AML1	6	6	6	12	11	10	9	13	14
BCR-ABL	8	7	6	13	12	12	22	18	15
MLL	7	5	6	10	9	8	13	14	18
Hyperdip>50	14	14	15	19	19	20	23	24	14
Ovarian	0	0	1	7	6	5	10	9	8
Prostate	9	9	8	10	9	10	14	10	8
Colon Tumor	14	11	12	12	10	12	12	10	12
ALL-AML	1	2	2	5	6	5	13	11	12
Subtype lymphoma	5	5	5	6	6	7	11	11	10
Stjude testing	12	7	6	14	12	9	26	22	19
Lung cancer	3	3	3	4	5	5	27	26	26
ALL-AML testing	4	4	3	3	4	4	3	3	3
Armstrong	0	0	0	2	2	2	0	1	1
Total Errors	85	75	75	119	113	111	185	174	162

feature number increases to 200. However, on the ALL-AML data sets, the error curve goes up when the feature number changes from 20 to 50, but goes down when the number changes to 100, and down again when it increases to 200. The error curves on the BCR-ABL and MLL data sets are even more fluctuated, showing contrasting shapes one another.

The situations for the other three classifiers are similar: The curve shapes are contrasting and fluctuated. This indicates that top 20 features can achieve the best performance only sometimes by a classifier on a certain application; top 50, 100, or 200 features can also achieve the best performance sometimes on some other data sets. We did not find a regular performance trend of the classifiers.

In term of total errors on the 15 data sets, SVM made 75, 72, 50, and 59 mistakes respectively when the top 20, 50, 100, and 200 features are used; these results are not better than its performance when the mean-entropy discretized features are used. Our CS4 classifier made 99, 83, 72, and 71 mistakes respectively when the top 20, 50, 100, and 200 features are used. The latter two results are close to the performance when the mean-entropy discretized features are used. As sometimes the number of mean-entropy discretized features is less than 200 or 100, our feature selection method shows advantage once again.

5.3 Wrapped Features, Comprehensibility and Stability

It is easy to understand our CS4 classifier. This classifier consists of a set of cascading decision trees; each tree is a set of rules; and each rule contains about 3 or 4 features. So, in a broad sense, CS4 is a set of rules that are organized by a cluster of decision trees. A typical rule is like the following: If *condition_1* = *true* and *condition_2=true*, then this sample is *positive*. Suppose a data set have n number of features, our CS4 classifier usually takes a small portion, sometimes very small, of the n features to construct the trees. So, in fact, CS4 conducts, in a wrapped manner, another round of feature selection. See Table 5 to compare the number of wrapped features used in CS4 and the total number of features of the data sets (original or reduced). For example, in the original ALL-AML testing data [11] where have 7129 features, CS4 uses only 31 features of them to construct the tree committee. Therefore the interpretation of the decision trees of CS4 does not necessarily require all the features but only the wrapped features to be involved.

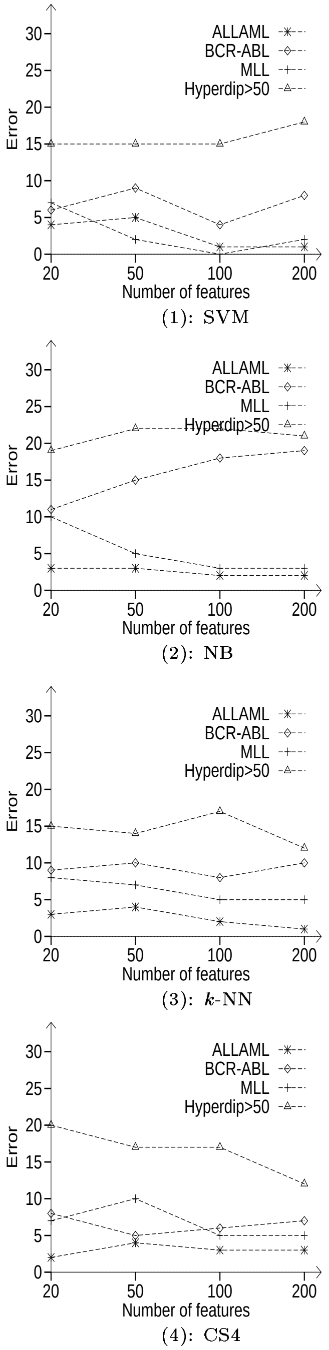
However, the interpretation of the learnt models of k-NN, SVM and NB must involve all the features. For example, in k-NN, the calculation of Euclidean distance between two points is the squared root of the sum of the squared differences at *all* the dimensions. A learnt SVM model is a non-linear function with variables equal to the number of the features. The extraction of rules from these models is difficult.

This model-structure difference between k-NN (or SVM, NB) and our CS4 classifier can help us to explain the stability of a classifier whether it is resistant to feature selection. Our observation is that if a classifier uses almost the same number of wrapped features before and after feature selection, then this classifier will be more or less resistant to feature selection. Otherwise, the classifier will be sensitive to feature selection as shown in k-NN, SVM or NB which must use all features for interpretation.

CS4 is an example of classifiers that are resistant to feature selection. From the last row, Total Errors, of Table 4, CS4 decreases its mistakes from 85, to 75, and maintains at 75 after the first round and the second round of feature selection. The normalized variance of the performance is as small as 0.0096. Correspondingly, the number of wrapped features in CS4 stands at the same level. For example, CS4 uses almost the same number —31, 26, or 25—features for classification on the original or reduced ALL-AML testing data; this also similarly occurs to the Lung-cancer data. (See Table 5.) For the sensitive classifiers such as k-NN, SVM, and NB, their normalized performance variance are 0.1987, 0.1236, and 0.3567 respectively, all much larger than CS4's 0.0096 indeed.

Table 5: CS4 wrapped features are far below the dimensionality of the data (original or after feature selection).

Data sets	Data dimension vs Number of wrapped features by CS4		
	original data	reduced data (1st round)	reduce data (2nd round)
ALL-AML testing	7129 vs 31	866 vs 26	350 vs 25
Lung-cancer	12533 vs 20	2173 vs 20	777 vs 20



(1): SVM

(2): NB

(3): *k*-NN

(4): CS4

Figure 2: Error numbers when selecting specific numbers of features

6. CONCLUSION

We have studied a new heuristics to select important features for classifying high-dimensional bio-medical data. There are two rounds of selection in the process. In the first round, our method ignores those features that are not discretized by an entropy method; In the second round, we use the mean-entropy values as filter to remove those less discriminating features. This feature selection method can significantly reduce the dimensionality of the problems, and meanwhile it can improve the performance of classifiers according to our experiments on 15 data sets.

We have also studied the resistance of a classifier to feature selection. Our proposed CS4 classifier is less sensitive to feature selection, compared to SVM, *k*-NN, and NB. Our reason is that CS4 is a decision-tree based classifier, it has a built-in method to select, in a wrapped manner, a small percentage of features from the whole feature space for the construction of the trees. So, even when the whole feature space is changed as by the feature selection methods, the number of wrapped features in CS4 is usually maintained. Thus, it should maintain its performance. The experimental results have also shown that CS4 is a highly accurate classifier.

7. REFERENCES

[1] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of National Academy of Sciences of the United States of American*, 96:6745–6750, 1999.

[2] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7:559–584, 2000.

[3] C. Blake and P. Murphy. The UCI machine learning repository. [`http://www.cs.uci.edu/~mlearn/MLRepository.html`]. In *Irvine, CA: University of California, Department of Information and Computer Science*, 1998.

[4] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[5] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[6] C. Cortez and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–279, 1995.

[7] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[8] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In A. Prieditis and S. J. Russell, editors, *Machine Learning: Proceedings of the Twelfth International Conference*, pages 94–202. Morgan Kaufmann, 1995.

[9] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In R. Bajcsy, editor, *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 1022–1029. Morgan Kaufmann, 1993.

[10] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In L. Saitta, editor, *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, Bari, Italy, July 1996. Morgan Kaufmann.

[11] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, October 1999.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer-Verlag, 2001.

[13] P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifier. In W. R. Swartout, editor, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 223 – 228. AAAI Press, 1992.

[14] J. Li, H. Liu, J. R. Downing, A. E.-J. Yeoh, and L. Wong. Simple rules underlying gene expression profiles of more than six subtypes of acute lymphoblastic leukemia (ALL) patients. *Bioinformatics*, 19:71 –78, 2003.

[15] J. Li, H. Liu, S.-K. Ng, and L. Wong. Discovery of significant rules for classifying cancer diagnosis data. *Bioinformatics*, 19:To appear, 2003.

[16] J. Li and L. Wong. Geography of differences between two classes of data. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery, PKDD 2002*, pages 325 – 337, Helsinki, Finland, 2002. Springer-Verlag.

[17] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA, 1993.

[18] E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing. Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, 1:133–143, 2002.