International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.8, No.12 (2015), pp.357-370
http://dx.doi.org/10.14257/ijsip.2015.8.12.33

Feature Selection Based on Dynamic Weight Boosting Algorithm

Cheng Zengping

Hubei University of Automotive Technology, ShiYan City, Hubei Province 442000, China

Abstract

Based on the problem that data simple weight is unchanged in current feature selection algorithms, we proposed a new boosting feature selection algorithm framework using data sampling technique to describe the change of data sample weight in the process of feature selection which can solve the problem that dynamic mutual information is sensitive to noise data. This method evaluates mutual information value among features by dynamic adjust data sample weights in the process of feature selection so that every selected feature can describe characteristics of data classification more precisely. Efficiency of this algorithm has been testified by experience results on UCI common test data sets.

Keywords: *Feature Selection, Mutual Information, Boosting, Weights*

1. Introduction

Along with the development of information techniques such as Internet or databases, lots of fields and industries produced and accumulated a lot of data. These data not only is huge in number, but also uses many features (or properties) to describe them. Knowledge means a form of data which has been processed and selected. It is vital in the process that human get to know and transform the world. The tradition method to get knowledge is directly building cognitive models from data by manual ways or institution so that we can get useful information people can understand. This manual way may seem workable and efficient facing small amount of data. However, when it turns to large even huge amount of data, it will come to its limitation and can't satisfy the needs of rapidly growing information. This is called the phenomenon of "Data explosion, lack of knowledge". Therefore, how to deal with the data effectively, find useful knowledge from massive data is the problem people facing now. And this is also the major research problem of information process.

As an interdisciplinary research field, knowledge discover (also known as data mining), machine learning and pattern recognition are all different displays of intellectual information processing techniques, where data classification is one of the specific topics or major directions of these research areas. Though there are many efficiency data classification learning algorithms which work well and can rapidly find out regularities among data facing little amount of data, we have new needs along with the emergence of new technologies. Data sets' accumulation and development towards large-scale ask current mining or learning method must adapt to this situation and can deal with, get useful knowledge from large amount of data rapidly. The meaning of large-scale data sets is mainly displayed in the following two ways: On the one hand, it refers to large sample (or instance) data sets themselves. On the other hand, dimensions used to describe features (or properties) of simples are quite high. For example, in text classification or information retrieval, every document is regarded as a simple and words or phrases in it are regarded as features; in image process, image is sample and pixels are features; in biology information, genes are regarded as features and protein made up by them is sample. As we can see, all these data or samples have a common feature: high dimension.

ISSN: 2005-4254 IJSIP

e.g., every document usually has thousands of words and phrases; every protein is usually made up by millions of genes.

The high dimension feature of sample data brings current mining or learning methods more needs and makes it more challenging because it not only influences the performance of learning algorithms, but also increases their complexity. In general, when learning method is trained on given data, there is a need that classification model's some unknown parses (such as probability density) have to be evaluated and modeled. These pareses are also related with the dimension of feature space. If the feature's dimension is too high, it would need lots of training samples to evaluate these parses. However, the amount of training samples is limited in real world. Therefore too high dimension of feature will lead to inaccuracy of parse estimation and finally affect the performance and efficiency of learning algorithm.

Usually, large-scale data sets contain lots of unrelated, redundant or useless features. Emergence of these redundant features not only increases feature space's dimension, decreases learning efficiency, but also increases noise data which will interrupt the process of learning or mining method and finally affect the construction of classification model. Therefore, in order to reduce these negative factors, decrease feature space's dimension, unrelated or redundant features should be removed from data so that we can decrease noise data's interruption, improve learning method's efficiency and performance and avoid over-fitting phenomenon when there isn't enough sample data. Dimensionality reduction is exactly an effective way to solve this problem. Dimensionality reduction is a multidisciplinary research field which contains statistics, databases, data mining, text mining, information retrieval, pattern recognition, image process, artificial intelligence, computational version and machine learning *etc*. Its main idea is that in the process of constructing classification model, we need to let learning method focus on those valuable or important features and ignore those unrelated or redundant features so that it can improve the efficiency of learning method and reduce computing complexity.

Dimensionality reduction can usually be achieved by two ways: feature selection and feature extraction. Feature selection is also called as variable selection or attributes selection. Different with feature extraction which changes original feature space, according to one kind of evaluation standard, feature selection will select an optimal or most efficient subset of features to replace original space which can achieve the purpose of reducing the dimension of feature space. By feature selection, unrelated or redundant features will be removed from original space and only those vital features will be retained. Feature selection is a classical research problem in statistics, machine learning and data mining. It is proposed to solve the problem of large-scale data computing.

Feature selection method can be regarded as a preprocessing step of classification algorithm as well as part of learning method. Based on its way combined with learning method, feature selection can be generally divided to three categories: embedded, filter and wrapper. Filter's way to select model is independent from classification method, which means it can be separately regarded as pretreatment step of classification learning method. Wrapper model treats feature selection method as part of learning method and directly uses classification performance as the evaluation standard of feature's weight. Embedded method will run feature selection and classification learning at the same time and find proper features in learning process.

In general, Filter model evaluates feature's importance by sample data's inner characters such as statistical correlation coefficient, mutual information and Fisher score *etc.* Liu [1-2] divided existing filter model evaluation standards into four categories which are distance standard, consistency standard, dependent standard and information standard. *e.g.*, in Relief [3] and its variety ReliefF [4], IRelief [5], Euclidean distance is used to evaluate feature subset's importance. Dash and Liu [6] use consistency factor to measure feature's distinguishing performance in sample classification. Wei and Billings [7] use squared correlation coefficient to evaluate every feature's weight in distinguishing

different categories. Abe and Kudo [8] use Bayes error bounds to select related features. Different with other three evaluation standards, information standard uses the concept of information entropy in information theory to quantify the magnitude of uncertainty between characteristics. Since it does not require assuming data distribution is prior known and can effectively measure the nonlinear relationship between features, information measurement draws a lot attention. Currently there are lots of algorithms proposed based on information entropy feature selection method such as Yu and Liu [9]'s using asymmetric uncertainty to measure relationship and redundancy between features. Feature selection method based on mutual information is quite sensitive to noise data. It's because in selection process, this method remove those samples can be recognized out from data sets after every cycle of feature selection and recalculate mutual information. If there is some noise data in sample data, these noise data would exist through all the selection process till the end. This also means along with the processing of this selection method, noise data's weight or its weight in mutual information evaluation will continue increasing causing this method selects redundant or unimportant features. In perfect model, people always hope the less noise data, the better because the less noise data there is, the more precise model we can get from learning method. Conversely, the more noise data there is, the more complex the learning process is and the more inaccurate, unreliable the model will be. However, in real world, noise data always exists and is unavoidable. Thus a reasonable solution is to reduce the adverse impact brought by noise data. Generally, there are two ways to solve noise data in machine learning field: sampling technique and sample weight.

Sampling technique means selecting part of sample data from data set based on some kind of selecting rules to represent the entire data set completing some specific activities like learning or constructing classifiers. Sample weight is another efficient way to deal with noise data. It assigns every sample in data set a weight prior learning or constructing model to reflect their importance in learning process. Different from sampling technique's reducing samples' amount method, sample weight method confirm sample data's influence on learning method by changing its weight. Sample's weight can usually be decided by two ways: One way is to preset it by experience and knowledge which is a reliable way but need lots of prior knowledge and manual interruption. So this way is obviously improper in large-scale data processing. The other way is to dynamically adjust or refresh sample weight during learning process to adapt to all kinds of learning atmosphere or destinations. Due to the high performance of models produced by it, this method is widely used in practice.

Boosting is a repeated sampling technique learning method which mainly uses Bootstrap technique to produce a serial of training sample data sets and use them to construct learning models, in which the later data set's sample distribution is produced according to the previous data set's classification errors [10]. In its specific learning process, single classification model is constructed in hierarchy way based on every sample weight data set and model's classification errors are used to refresh sample's weight to construct another classifier. These separate classification models finally aggregate into a more powerful classification model. Through this learning way, a week classifier performing slightly better than random guessing will be updated to strong classifier with few classification errors [11]. Due to such excellent performance, boosting learning method is concerned much by researches in machine learning field since it is proposed and is one of the main research directions and hotspots.

Currently, most existing feature selection methods' sample weight keeps unchanged which means every sample keeps unchanged during selection process and has the same weight. This means once sample data sets are provided, then value of evaluation standard is settled and noise data holds the same weight as correct data. Obviously, this is unreasonable because this kind of evaluation standard can not accurately reflect the characteristic that the degree of correlation between features will dynamically change

along with the feature selecting process. To solve this problem, this paper proposed a new feature selection algorithm framework based on boosting learning method. This method will automatically adjust sample weight after every feature selection cycle and recalculate feature's evaluation standard to make sure it can truly reflect degree of correlation between features.

2. Related Works

2.1. Feature Selection

Research on feature selection started in 1960s, which was mainly from statistical point of view at that time and didn't involve high dimension. Along with the emergence of new techniques and the development of machine learning, feature selection draw more and more attention in machine learning field which promoted a new round wave of feature selection. Besides, the emergence of large scale data also brought a serious challenge to traditional feature selection. Though feature has been widely used, there is still no uniform or perfect definition of it since feature selection focus on different points based on different problems and needs. There is a sample data set $T = (O, F, C)$ in which $F = \{f_1, f_2, ..., f_m\}$, $C = \{c_1, c_2, ..., c_k\}$, $O = \{o_1, o_2, ..., o_n\}$ represent features, classifications and sample data sets. Let $J : 2^F \rightarrow [0,1]$ to be the evaluation function for feature subset in which the bigger J(X) is, the more information feature subset X has. In this case, feature selection generally has such following three types: (1) Find a feature subset X from feature sets F to make J(X) biggest. (2) Set a threshold J0 and find a smallest subset X from F to fulfill the requirement of J(X)>J0. (3) Find a subset X from F to make J(X) as bigger as possible and make features' in X as fewer as possible. These three ways reflect different aspects and focuses of feature selections. The first way focuses on selected subset's information amount which means this selection method will try its best to loss less information. The second way emphasizes selecting a smallest subset fulfilling given conditions. And the last method tries to find fit compromise between subset size and information amount. We can see from this that feature selection plays a vital role for classification learning method cause the quality of its choosing subset will directly determine classification model's final performance. On the one hand, a good feature subset can obviously improve classification learning method's efficiency, its classification model's performance and even to some extend improve its generalization ability. So it can help to effectively avoid noise data's interference. On the other hand, classifier's performance also reflects the quality of selected subset which means a perfect subset should contain as much classification information as possible. Therefore, in classification process, feature selection method usually selects those subsets which can provide high classification performance and has as fewer features as possible.

Generally speaking, feature selection is consisted up by four steps: initial subset set, searching strategy, and subset evaluation and termination condition. Initial subset set is the start of feature selection method as well as the beginning of searching process. Its selected results directly influence later searching strategy. In general, if initial subset S is null, which means this method doesn't have selection feature at the beginning, later searching process will add candidate features to selection subset step by step which is called forward search. If initial subset is original feature set, which means S = F, then searching process will continue to delete redundant or related features from selection subset S which is called backward search. And if initial subset is randomly produced from feature set F, then searching process will adopt random searching strategy to add candidate features or delete selected features. Termination condition is to decide, based on candidate subset S's evaluation value J(S) or other constraints, whether current candidate subset S meets preset conditions. If it meets, then the selection method will end and return candidate feature subset S as final results. If not, the searching process will

cycle to form new candidate subsets until the termination condition is met. There are some termination conditions usually used in feature selection method: 1) Amount of candidate subset S's features exceeds preset threshold. 2) Time of searching cycle exceeds preset threshold. 3). Value of evaluation function J(S) gets to biggest or best performance. 4). Value of evaluation function J(S) exceeds preset threshold. Among these four steps, searching strategy and evaluation standard are two most important problems. A good searching strategy can accelerate choosing speed and find best results. A good evaluation standard can ensure all subsets contain rich information, reduce error selection and improve algorithm's performance.

The process of feature selection in some extend is a problem of searching optimal subset in which every candidate subset S is a state of searching space. In its specific process, the generation of candidate subset, which is searching direction, is one of problems must be taken into consideration. It has four ways: Forward searching which will add one or more new features to current candidate subset. Backward searching which will remove one or more features from current candidate subset. Bi-direction searching which will remove some features first and then adds some new features. Random searching which will form a randomly candidate subset. The first three searching ways usually take greedy strategy to select candidate features need to be added in or removed from subset which means adding features with best performance among candidate features to current candidate subset S or removing features with worst performance in candidate subset S. By doing this, every step of searching process will move towards best direction to find optimal results. Evaluation standard is mainly a kind of evaluation way based on some measurement standards to evaluate the quality of selected features and subsets. Since evaluation standard will directly determine the output of selection method and the performance of classification model, it plays an important role in feature selection method. What's more, the same feature selection method with different evaluation standard may generate different "optimal" feature subsets. Therefore, evaluation standard is always the research hot topic of feature selection method. Up to now, lots of evaluation standards have been proposed which can be divided into five categories: Distance standard, Consistency standard, Dependent standard, Information standard and Classification error standard.

2.2. Boosting

Based on approximately correct probability (PCA) learning model, boosting learning method is a common method proposed to improve performance of given learning method [12]. In PCA learning model, if a set of concepts can be recognized by a polynomial level learning method with high accuracy, they belong to strong learnable methods. Otherwise, if they can only be recognized by a learning method slightly better than randomly guessing, they belong to weak learnable methods. PCA learning model is focusing on whether we can find a learning method to improve weak learning method to strong learning method [13]. This problem is the basement of other computational learning theory such as statistic and ensemble learning. Since if there is such method to improve performance, there is no need to pursue those hard-to-find and complex strong learning methods. Instead, some common, simple weak learning methods will be enough. To solve this problem, Schapire gives a certain answer by using a constructing method in paper [11] which says there are some ways to improve weak learning method to strong learning method. This constructing process is called boosting method.

Boosting method is a learning construction process. It first serially uses simple, not quite right, experience based weak learning methods to get many classifiers, in which the K times learning process rely on results of K-1 times learning before. And then combine these classifiers together based on some kind of rules to finally get a learning method with high accuracy and complexity. In its K times learning process, boosting method will focus on learning of those samples which produce errors in K-1 times training before.

This process in similar to the process of points of knowledge's learning or review in teaching. We first learn through all points of knowledge. For those points of knowledge easy to be remembered, we will catch them in the first time, and for those points of knowledge hard to understand, we will focus on them in the second time. And in the third time, we will focus on those points haven't been understood in the second time. This learning process will continue until all points of knowledge have been totally understood. Though it can be proved in theory that boosting method can combine lots of weak classifiers together to get a model with strong classifier's generation ability, first proposed boosting method had one obviously shortage that it requires prior knowledge of weak learning method's lower limit performance. There is a new adaptive boosting method proposed by Freund and Schapire later which is called AdaBoost[14] which can solve this problem. For its high efficiency and performance as well as boosting theory's solid basement, AdaBoost has now become one of two famous learning methods in machine learning along with SVM method and also is the most widely used boosting learning method. Boosting method has complete, solid theory foundation and can achieve any accuracy degree by given weak classifiers having enough training data. Compared with other learning methods, boosting method has the advantages such as simple, effective and adaptable. Besides, it doesn't need any more parses except the number of iterations. Also, it can combine with any weak learning method such as decision tree or artificial neural networks and has been applied in many application fields such as nature language understanding, image retrieval, test classification *etc*. Boosting method also doesn't need prior knowledge of weak learning methods. And with enough training samples, it can finally produce a strong classifier with high performance by constructing a weak classifier achieving the classification goal of high efficiency and performance [15]. Having so many advantages, boosting method is an efficient tool improving weak learning method's performance and is one of the most efficiency learning methods in recent machine learning field.

3. Feature Selection Based on Dynamic Weight

Mutual information is usually used as evaluation standard of feature selection method for the reason it can effectively measure the degree of feature's correlation. It can be known from its definition that the basement of mutual information is statistic theory. In real situations, once training sample data set is given, the statistic distribution of every feature is also settled down. This means the degree of correlation or mutual information between features and categories is settled down and will contain unchanged through whole feature selection process. It is inappropriate in some extend since it will cause one problem that mutual information can not accurately reflect the whole dynamic process of feature selection.

3.1. Weight Updating

While constructing classifier, if every time after adding features it can accurately recognize more samples, it will have better performance and stronger generation ability. This proves that in the process of feature selection, selection algorithm method always hope selected features can recognize as much unrecognized samples as possible without paying any attention to those samples already been recognized, whose knowledge has been contained by selected features. In other words, in the same training sample set, every sample plays different roles for candidate features. And along with this selection process, sample weights should also change adaptively. Generally, those samples haven't been correctly recognized should have bigger weight than those have been correctly recognized. Only by this way, it can be insured that features which are going to be selected will pay more attention to those unrecognized samples. When unrecognized samples are recognized by new selected feature, their weight should be decreased.

Unfortunately, most current feature selection algorithms haven't taken this point into consideration.

Boosting method has an excellent performance on the process of sample weights. It is firstly proposed to improve weak classifier learning method's performance in machine learning field [10]. With solid theory basement and excellent performance, boosting method has drawn a lot attention in many fields such as model recognition and statistics. Using weak learning methods as input, it serially constructs a set of weak classifiers in training process and finally combines these weak classifiers in a liner way (*e.g.*, primary voting or average voting) into a strong classifier. The main idea of boosting method is paying special attention to those samples hard to be classified or recognized which means increasing those unrecognized samples' weight after every time of training.

Based on the weight updating thought in boosting learning method, we adopt similar updating way to achieve adjusting sample weights dynamically in feature selection environment. First, all samples in training sample set are initialized a same weight value. Then, along with the running of feature selection process, weight of those features which can be correctly recognized will be decreased and weight of those features which can't be recognized will be increased so that next round of selection will pay more attention to those unrecognized samples. This will help following selected features to recognize more unrecognized samples. After adjusting sample weights, the degree of correlation between candidate features and classification categories will change so that measure standard can accurately reflect the dynamic changing process of related degree during selecting process. This can prove that this feature selection process can recognize as many samples as possible.

3.2. The Proposed Algorithm

Based on analysis above, here we proposed a new boosting feature selection algorithm framework which uses mutual information as evaluation standard of correlation degree between features. Constructor of boosting feature selection algorithm framework is shown below as algorithm one in which δ is the amount of features needed to be selected preset by users.

Input: A training dataset T = (O, F, C), where |O| = n;
The iterative times δ;

Output: A feature subset S;

1). Initialize relative parameters, *e.g.*, $S = \Phi$, $i = 1$;

2). Set the weight $w_{1,j}$ as 1/n for each sample x_j in T, where j = 1, ... , n;

3). While $i <= \delta$ do

4). Calculate the information criterion J(f) for each feature f in F;

5). Select the feature f with maximal J(f);

6). Insert f into S and remove from F, *i.e.*, S = S + {f}, F = F - {f};

7). Obtain the error e of f or S with T ;

8). Update the weight $w_{i,j}$ for each sample x_j in T according to the error function g(e), and then normalize them;

9). i = i + 1;

10). End while

11). Return the subset S as the selected features.

Algorithm one's working theory is quiet simple. It first initializes related pareses and assigns all samples' weight the same value which is 1/n (n is the amount of samples in data set). Then this algorithm will step to feature selection cycling stage. In every cycle, algorithm first calculates every candidate feature's evaluation standard value J(f) and then chooses the one with biggest J(f) value to be candidate feature put into selected subset S. After that, this method will calculate error rate e on this training sample set from features f or selected set S. Based on error function g(e), it will update every sample's weight to increase weights of unrecognized samples. After the updating of sample weights, they still need to be normalized for next cycle of feature selection. All this cycle process will last until amount of selected features exceed preset thresholdδ. The perfect value ofδis hard to set. Ifδis too small, feature subset we finally get will lose lots information. If it is too big, there will be some redundant or useless features making this feature selection meaningless. One comprise solution is using inconsistent factor of sample data set as termination condition of this selection cycle. The advantage of this solution is that selection method can adaptively set proper selection amount.

Besides thresholdδ, boosting algorithm also needs three more main factors: evaluation function J(f), way to get error value e and refreshing strategy of sample weight. Experiment following uses mutual information as evaluation standard to measure degree of correlation between features. Most current boosting learning methods' error rate e is obtained from the performance of embedded interior base classifiers (weak learning method). In subsequence simulations, we use Bayesian formula to calculate error rate e due to its relatively high performance and best estimated error rate [14]. The third factor to be considered in algorithm 5.2 is the updating of sample weight which mainly contains two parts: error function g(e) and updating strategy. Error function is mainly used to improve or reduce sample weight and directly influences the performance of algorithm which means selection of this function play a vital role in this algorithm. Traditional boosting learning method usually sets error function g(e) as (*i.e.*, $g(e) = (1-e)/e$), in which error rate e<0.5. On the other hand, updating strategy of sample weight mainly deal with the problem which sample weights should be increased and which sample weights should be decreased. In general, there are three updating strategies usually used. Strategy one: this strategy mainly work on situation that sample data going to be updated only have direct relationship with current selected feature f. Strategy two: this strategy is similar to strategy one. They both multiply g(e) with correctly recognized samples' weight and divide unrecognized samples' weight by g(e). The difference is that it does not only take consider of current selected feature f, but also uses former selected features' information. Strategy three: the main idea of this updating strategy comes from primary voting. In this updating method, every sample in sample data set will change based on whether it can be recognized by over half selected features. If it can, then its weight will be decreases by g(e). Or its weight will be increase by 1/g(e). Following simulations will compare performances of this three updating strategies.

4. Experiments and Analysis

In this chapter, we will compare proposed boosting feature selection framework with three typical feature selection algorithms BIF [16], mMIFS-U [17] and BDSFS [18] to vertify its performance. The reason to select these three algorithms is that they all have relatively high efficiency and performance. BIF and mMIFS-U methods are two feature selection algorithms based on information measurement. Choosing those features with biggest mutual information value, the evaluation standard of BIF method is mutual information. mMIFS-U method is a improved MIFS-U algorithm proposed by Novovicova [17]. BDSFS is a typical boosting feature selection method whose base classifier is DecisionStump. As mentioned before, the final result of BDSFS is an integrated classifier but not feature subset.

4.1. Test Dataset

To compare the performance of algorithm with other algorithms comprehensively, simulations used ten different scale UCI common test data sets. These sample data sets all come from UCI machine learning repository [19] which is usually used in papers to check the performance of classification learning methods or feature selection methods in machine learning or data mining fields. There are some simple descriptions about these test data sets in Table 1 such as name, sample amount, feature amount and classification amount. More information about data sets can be found in UCI machine learning website. As we can see in Table 1, these data sets contain different sample amount, feature amount and classification amount. The amount of samples is between 355 and 8124 and the amount of features is from 22 to 216. The diversity of these data sets can prove that they can to some extend display algorithm's efficiency in different conditions.

Table 1. Summary of Experimental Data Sets Described

	Dataset	Number of sample	Number of feature	Number of categories
1	Anneal	898	38	6
2	Cylinder-bands	540	36	2
3	Hypothyroid	3772	39	4
5	Ionosphere	355	34	2
5	Kr-vs-kp	3196	36	2
6	Mfeat-factors	2000	216	10
7	Mfeat-zernike	2000	47	10
8	Mushroom	8124	22	2
9	Musk clean	476	166	2
10	Spectrometer	531	100	4

4.2. Experiment Setting

After pre-processing of sample data set, it will be treated as the input parse of feature selection algorithm. To be even, every feature selection algorithm will choose same amount of features in classification performance test. As mentioned before, inconsistent factor can be used as the termination condition of feature selection cycle to adaptively end this selection process. What's more, when it is used as termination condition, amount of selected features would also be a little fewer. Therefore, we uses inconsistent factor as termination condition of selection process cycle. Specifically, selection algorithm first calculates sample data set's inconsistent factors under all features' conditions. Then it uses BIF algorithm to do feature selection and calculate sample data set's inconsistent factor under selected subset S after every feature selection. If this new factor is basically the same as initial factor, the BIF algorithm will end and return selected feature subset. At this time, amount of selected subset's features will be input of other selection algorithm representing amount of features needed to be selected. Simulation was run on computer whose CPU is 2.8GHz clocked Pentium IV and memory is 512MB.

To be even, all feature selection algorithms were coded by MS VC++ 6.0. The testing environment of simulations is Weka [20] software and all related parses in this simulation was set as default value. To get reliable results, simulations adopted 10 times cross-validation method on verification of classification performance and repeated it three times to get average value as final result. Besides, statistical t-test was also used to testify whether feature selection method can obviously improve or reduce performance of learning method in statistic. In this simulation, except BDSFS method which is Wrapper model, all other three algorithms belong to Filter model which means they are irrelevant with specific learning method. So simulation also needs extra learning method to verify selected subsets' performance in classification model. For the reason that single classification model may be good at one specific feature selection algorithm, our

simulations used two typical classification learning methods, which are naive Bayesian classifier (NBC) and nearest neighbor learning method (1NN), to verify algorithms' performance avoiding over-fitting.

4.3. Experimental Results

We will introduce four sets of experiments in this chapter to comprehensively verify the performance of boosting selection method. These four sets of experiments are used to compare the performance between boosting methods and un-boosting methods, performance between boosting methods and different influences on boosting selection methods' performance by different updating strategies or error functions.

First set of simulations mainly compare the performance of un-boosting methods BIF and mMIFS-U with their corresponding boosting methods BBIF and BmMIFS-U in 1NN and NBC classifiers in which BBIF and BmMIFS-U methods are separate achieves of BIF and mMIFS-U methods in algorithm's framework. Evaluation standards of BBIF and BIF are the same which are both mutual information. Evaluation standard of BmMIFS-U is also the same as mMIFS-U. Table 2 gives out the comparison of those four feature selection methods' classification performance in NBC learning method in which error function g(e) of BBIF and BmMIFS-U those two boosting methods used $g(e) = 1/(1-e)$ while updating strategy is strategy two which considers all selected features' information. The "original" column in table 2 indicates NBC's classification accuracy on original feature space. We can see from data in table 2 that BBIF's performance is obviously better than BIF. *e.g.,* none of BBIF's classification performance in these 10 sample data sets is lower than BIF. BBIF obviously improved NBC's classification performance in four data sets while BIF didn't make any obvious improvement. Though BBIF decreased the performance of NBC in three sample data sets, BIF did worse on these three sample data sets. Besides, BIF obviously decreased NBC' performance on Spectrometer data set (Line 10). Also we can see, BBIF and BIF perform same on data set Anneal (Line 1) and Hypothyroid (Line 3). It's because these two feature selection method selected same features on these two data sets. Similar situations can be found from performance comparison between BmMIFS-U and mMIFS-U selection methods. *e.g.,* BmMIFS-U's accuracy on 9 data sets is not lower than mMIFS-U. And for data set Kr-vs-kp(Line 5), though BmMIFS-U's classification performance is not as good as mMIFS-U, its absolute value of difference of performance compared with original data is quite little. While mMIFS-U obviously lowered NBC's performance on six data sets, BmMIFS-U just reduce three data sets' performance. Besides, BmMIFS-U and BBIF's average performance is also better than mMIFS-U and BIF.

Table 2. Performance Comparison between BIF, mMIFS-U and their Corresponding Boosting Method in NBC Classifier

	Original	BIF	BBIF	mMIFS-U	BmMIFS-u
1	96.12	92.34	92.34	93.23	93.39
2	74.07	74.87	75.13	71.81	73.77
3	98.61	98.51	98.51	96.97	97.68
4	90.75	90.75	92.19	90.26	91.81
5	87.79	87.93	90.33	88.81	87.57
6	93.91	85.48	88.33	77.72	87.26
7	74.30	65.55	69.87	68.24	70.85
8	95.51	94.39	99.21	94.39	97.93
9	86.26	87.43	92.22	87.16	88.43
10	59.51	56.73	60.13	56.60	58.61
AVG	85.68	83.41	85.83	83.51	84.74

Table 3 is BIF, mMIFS-U and their corresponding boosting methods' performance in 1NN classifier. From results on the table, we can conclude that boosting method and un-boosting methods' performance is similar in 1NN to that in NBC that BBIF's performance is obviously better than BIF's and BmMIFS-U is also better than mMIFS-U. *e.g.,* None of BBIF and BmMIFS-U's classification accuracy in those 10 data sets is lower than their corresponding un-boosting methods. For the reason that BIF and BBIF selected same features in Anneal (Line 1) and Hypothyroid (Line 3), their corresponding performance is the same. Similar with this situation in NBC, these four methods all obviously decreased learning method's performance on Mfeat-factors and Mfeat-zemike. The reason is that amount of selected features is too small like BIF just selected 8 features from Mfeat-factors' 216 features.

Table 3. Performance Comparison between BIF, mMIFS-u and the Boosting Methods in 1NN Classifier

	Original	BIF	BBIF	mMIFS-U	BmMIFS-u
1	93.32	93.09	93.09	93.15	93.39
2	76.29	74.64	75.13	74.02	75.02
3	97.93	97.98	97.98	98.16	98.57
4	92.87	92.75	92.87	92.01	92.75
5	90.36	94.37	96.25	93.70	94.78
6	95.57	81.61	86.28	74.34	84.39
7	70.57	62.06	65.58	66.39	67.36
8	100.0	99.83	100.0	99.81	99.99
9	89.71	85.67	89.34	86.58	88.25
10	59.83	56.73	61.83	58.18	58.75
AVG	86.65	83.88	85.84	83.63	85.31

1. Performance Comparison between Boosting Methods

To verify the fact that proposed method performed better than other boosting methods, the second set of experiments mainly compare BBIF, BmMIFS-U and BDSFS methods' performance on these data sets. Their experiment environments are the same as the first set which means they took the same error function and updating strategy. As we should notice, in original BDSFS method, amount of selected features is preset by users. To be even, there was some change to make its selected features are same as BBIF and BmMIFS-U's. Specific experimental results is shown in Table 4 in which bold value represents that it is highest in NBC or 1NN. As we can see from the experimental result, BBIF has the best performance in these three selection methods. Its classification performance is better than BmMIFS-U and BDSFS in most cases. *e.g.,* BBIF's classification performance in NBC learning method is best in 8 data sets while none of BDSFS's performance is highest. Though BDSFS in 1NN performed best in Cylinder-bands data set (Line 2), the absolute value of difference between its classification performance and BBIF's or BmMIFS-U's classification performance is not large. What's more, BmMIFS-U performed best in both Anneals (Line 1) and Mfeat-zernike (Line 7) data sets.

Table 4. Performance Comparison between BBIF, BmMIFS-U and BDSFS in NBC and 1NN Classifier

	NBC			1NN		
	BBIF	BmMIFS-U	BDSFS	BBIF	BmMIFS-U	BDSFS
1	92.34	93.39	92.33	93.08	93.40	93.08
2	75.12	73.76	74.95	75.14	75.01	75.39
3	98.53	97.67	97.53	97.98	98.58	96.54
4	92.20	91.80	90.11	92.89	92.75	92.37
5	90.34	87.56	87.81	96.25	94.78	94.79
6	88.35	87.25	85.43	86.30	84.35	81.99
7	69.85	70.86	65.17	65.57	67.34	60.84
8	99.23	97.92	98.41	100.0	99.99	99.99
9	92.22	88.42	82.67	89.34	88.24	84.45
10	60.13	58.62	59.95	61.83	58.75	60.97
AVG	85.84	84.72	83.45	85.84	85.31	84.04

To clearly compare these three kinds of methods, we calculated average values of their performance in NBC and 1NN. Also we used statistical t-test to calculate average performance difference of BBIF, BmMIFS-U and BDSFS. Results are shown in Figure 1 in which horizontal axis represents data sets and vertical axis represents statistic p value. Bar graph located above or below horizontal line separately represents corresponding methods are better or worse than BDSFS. What's more, while the absolute value of p is bigger than 2, it represents corresponding methods have better or worse average performance than BDSFS. As we can see from Figure 1, except for the first two data sets, BBIF has a higher accuracy than BDSFS while BmMIFS-U performes worse on the second, fifth, eighth and tenth data sets compared with BDSFS. BBIF's average performance is only worse than BmMIFS-U on the first and seventh data sets. Though BDSFS has the best performance on data set 2, their absolute difference value on this data set is no more than 0.78%. In one word, boosting method proposed in this paper has better performance than BDSFS.

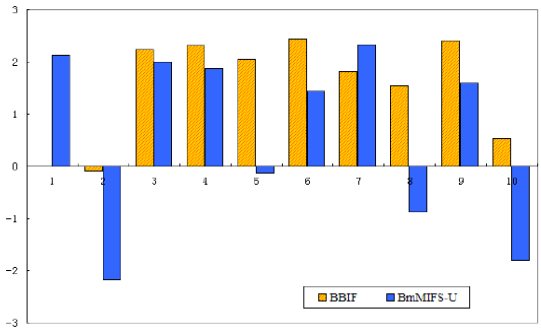

Figure 1. Statistic Performance Comparison among BBIF, BmMIFS-U and BDSFS

2. Updating Strategy

The third set of experiments is used to verify the conclusion which is the selection of updating strategy will directly influence the performance of boosting method. This set of experiments used three kinds of updating strategies under the error function $g(e) = 1 / (1 - e)$ in BBIF method. Its specific experimental results are shown in table 5. As shown by result data, different updating strategies will make different influences on

boosting method's performance. Even for the same data set, their differences are quite large. In general, strategy 2 is better than the other two strategies while strategy 1 is little worse than strategy 3. *e.g.*, Strategy 2 (S2) performs worse than strategy 3(S3) in no more than three situations in NBC and 1NN while all its performance in every situation is better than strategy 1(S1). The reason strategy 1 performs worst is that it doesn't take full advantage of selected features' information.

Table 5. Performance Comparison of BBIF among Three Strategies in NBC and 1NN

	NBC			1NN		
	S1	S2	St3	S1	S2	S3
1	92.34	92.33	92.34	93.09	93.09	93.09
2	74.69	75.14	74.94	74.45	75.15	75.02
3	98.51	98.51	98.51	97.99	97.99	97.99
4	92.48	92.19	93.05	92.19	92.86	92.68
5	88.05	90.34	89.06	94.26	96.23	96.45
6	86.35	88.34	86.70	84.38	86.29	85.15
7	65.13	69.81	65.54	62.32	65.59	62.06
8	97.33	99.22	98.91	99.46	100.00	99.46
9	85.83	92.23	85.89	88.63	89.34	87.83
10	56.66	60.14	60.32	55.79	61.81	62.65
AVG	83.73	85.84	84.52	84.25	85.84	85.24

3. Error Function

As mentioned before, besides updating strategies, error function g(e) is another vital factor to evaluate boosting method's performance. The choosing of g(e) will also directly influence boosting method's performance. Therefore, the fourth set of experiments mainly verifies error function g(e)'s influence on boosting method by testing three different error functions' classification performance in BBIF in which error updating strategy was strategy 2. These three error functions are index, division and reciprocal form of error e which are $g1(e) = \exp(1/(1-e))$ $g2(e) = (1-e)/e$ and $g3(e) = 1/(1-e)$. Table 6 gives out performance comparison of these three functions in NBC and 1NN methods.

Table 6. Performance Comparison of BBIF Method among Three Error Functions in NBC and 1NN

	NBC			1NN		
	g1(e)	g2(e)	g3(e)	g1(e)	g2(e)	g3(e)
1	92.03	92.03	92.34	92.28	92.28	93.09
2	74.58	74.58	75.13	76.39	73.66	75.14
3	98.58	98.51	98.51	97.90	97.67	97.98
4	90.18	90.95	92.18	92.0	92.00	92.88
5	86.57	86.70	90.33	94.63	93.20	96.25
6	88.55	87.11	88.33	86.89	85.20	86.27
7	71.04	69.86	69.86	67.93	65.57	65.57
8	98.71	98.40	99.22	99.74	99.99	100.0
9	88.84	7.08	92.23	90.11	89.17	89.32
10	60.45	60.16	60.12	62.09	61.20	61.83
AVG	84.95	84.54	85.82	86.00	85.00	85.83

5. Conclusion

Based on the problem that current feature selection methods' information metrics standard can't correctly reflect degree of correlation, we proposed a new boosting feature selection framework. It measures features' correlation by dynamically adjusting sample weights and can accurately describe data classification's characteristic. Then we discussed about updating strategy of sample weight and error function in proposed method. To verify performance of this method, simulations were done in ten UCI common test data sets. Experimental results show that boosting method proposed in this paper performs better than its corresponding un-boosting method and typical boosting method BDSFS in most cases. We can also see from these results that error function plays a vital role in boosting method. None of three error functions adopted in this paper is obviously better than the others which mean they all have their own advantages. Therefore, one of future's main works is to find better error function or get best performed error function by integrated way.

References

[1] H. Liu and H. Motoda, "Feature Selection for Knowledge Discovery and Data Mining [M]", Boston: Kluwer Academic Publishers, **(1998)**.
[2] H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering", IEEE Transactions on Knowledge and Data Engineering, vol.7, no. 4, **(2005)**, pp. 491-502.
[3] K. Kira and L. Rendell, "A practical approach to feature selection", Proc of the 9th International Conference on Machine Learning, **(1992)**, pp. 249-256.
[4] I. Kononenko, "Estimation attributes: analysis and extensions of RELIEF", Proc of the European Conference on Machine Learning, Catania, Italy, **(1994)**, pp. 171-182.
[5] Y. Sun, "Iterative RELIEF for Feature Weighting: Algorithms", Theories, and Applications, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, **(2007)**, pp. 1035-105.
[6] M. Dash and H. Liu, "Consistency-based search in feature selection", Artificial Intelligence, vol. 151, no. (1-2), **(2003)**, pp. 155-176.
[7] H-L Wei, S A. Billings, "Feature Subset Selection and Ranking for Data Dimensionality Reduction", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no .1, **(2007)**, pp. 162-166.
[8] N. Abe and M. Kudo, "Non-parametric classifier-independent feature selection", Pattern Recognition, vol. 39, **(2006)**, pp. 737-746.
[9] L. Yu and H. Liu, "Efficient Feature Selection via Analysis of Relevance and Redundancy", Journal of Machine Learning Research, vol. 5, **(2004)**, pp. 1205-1224.
[10] Y. Freund and R. Schapire, "Experiments with a New Boosting Algorithm", Proc of the 13th International Conference on Machine Learning, Bari, Italy, **(1996)**, pp. 148-156.
[11] R E. Schapire, "The Strength of Weak Learn ability", Machine Learning, vol. 5, no. 2, **(1990)**, pp. 197-227.
[12] L G. Valiant, "A theory of the learnable", Communications of the ACM, vol. 27, no. 22, **(1984)**, pp. 1134-1142.
[13] M. Kearns and L G. Valiant, "Learning Boolean formulate or finite automata is as hard as factoring Number", Aiken Computation Laboratory, Harvard University, **(1988)**.
[14] Y. Freund and R E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting [J]", Journal of Computer and System Sciences, vol. 55, no. 1, **(1997)**, pp. 119-139.
[15] P. Buhlmann and T. Hothorn, "Boosting algorithms: regularization, prediction and model fitting [J]", Statistical Science, vol. 22, no. 4, **(2007)**, pp. 477-505.
[16] A K Jain, R P W Duin and J. Mao, "Statistical Pattern Recognition: A Review [J]", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, **(2000)**, pp. 4-37.
[17] J. Novovicova, P. Somol and M. Haindl, "Conditional Mutual Information Based Feature Selection for Classification Task", Proc of the 12th Iberoamericann Congress on Pattern Recognition, Valparaiso, Chile, November, **(2007)**, pp. 417-426.
[18] S. Das, "Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection", Proc of the 18th International Conference on Machine Learning, San Francisco, CA, USA, **(2001)**, pp. 74-81.
[19] A, Asuncion and D J. Newman, "UCI Machine Learning Repository [EB/OL]", Irvine: School of Inf and Comp Sci, Univ of California, http://www.ics.uci.edu/~mlearn/MLRepository.html, **(2007)**.
[20] I H. Witten and E. Frank, "Data Mining-Pracitcal Machine Learning Tools and Techniques with JAVA Implementations", 2nd ed. CA: Morgan Kaufmann, **(2005)**.

