Bilkent University
Computer Engineering Department
Technical Report No: BU-CE-0901

Voting Features based Classifier with Feature Construction and its Application to Predicting Financial Distress

H. Altay Güvenir[1], **Murat Çakır**[2]

[1] Bilkent University, Computer Engineering Department, Ankara, Turkey,
[2] Central Bank of the Republic of Turkey, Ankara, Turkey

Corresponding author: H. Altay Güvenir
Bilkent University
Computer Engineering Department
06800 Ankara, Turkey
E-mail: guvenir@cs.bilkent.edu.tr
Tel: +90 312 290 1252
Fax: +90 312 266 4047

Abstract

Voting Features based Classifiers, shortly VFC, have been shown to perform well on most real-world data sets. They are robust to irrelevant features and missing feature values. In this paper, we introduce an extension to VFC, called Voting Features based Classifier with feature Construction, VFCC for short, and show its application to the problem of predicting if a bank will encounter financial distress, by analyzing current financial statements. The previously developed VFC learn a set of rules that contain a single condition based on a single feature in their antecedent. The VFCC algorithm proposed in this work, on the other hand, constructs rules whose antecedents may contain conjuncts based on several features. Experimental results on recent financial ratios of banks in Turkey show that the VFCC algorithm achieves better accuracy than other well-known rule learning classification algorithms.

Keywords:

Classification, voting, feature construction, financial distress, feature projections

1 Introduction

Voting Features based Classifiers, shortly VFC, have been shown to perform well on most real-world data sets. The VFC previously developed, e.g., CFP (Güvenir, & Sirin, 1996), VFI (Güvenir, Demiröz, & Ilter, 1998), BCFP (Güvenir, Emeksiz, Ikizler, & Örmeci, 2004), learn a set of rules that contain a single condition based on a single feature in their antecedent. Given a query, each feature, based on the value of the query instance for that feature, distributes its vote among possible classes. The class that receives the highest amount of votes is declared as the predicted class label of the query instance.

The basic CFP (Classification by Feature Partitioning), VFI (Voting Feature Intervals) and BCFP (Benefit maximizing Classifier on Feature Projections) algorithms have been shown to perform quite well on most real world data sets, including some of the ones in the UCI Repository (Asuncion, & Newman, 2007). They are shown to be robust to irrelevant features and missing feature values (Güvenir, 1998). CFP employs an incremental approach to learning the model. It partitions the feature values into segments that are generalized or specialized as the training instances are processed. The VFI, on the other hand follows a non-incremental approach in forming a set of feature intervals, which represent either a range of feature values, or a point for single feature value. During the training period of VFI, the end points, i.e., the minimum and maximum values, for each class on each feature dimension are determined. The list of end points on each continuous feature dimension is then sorted. If the feature is nominal, each distinct end point constitutes a point interval. Each of the intervals on each feature forms a classification rule. BCFP algorithm also uses a non-incremental learning approach. However, given a benefit matrix, it learns classification rules that maximize the benefit of classification. In the querying phase, using these rules, the BCFP algorithm tries to make a prediction maximizing the benefit.

The way the VFC algorithms learn a model and use it for classification is illustrated in Fig. 1a. This simple data set contains four training instances represented by two features; one of them is nominal (f_1) and the other is continuous (f_2). The class labels are A and B.

The model learned contains two rules on each feature. A rule has a vote of 1, and it distributes that vote among the possible class labels in the given domain. The rules for f_1 are:

If f_1 = a Then vote[A]=1.0, vote[B]=0.
If f_1 = b Then vote[A]=0, vote[B]=1.0.

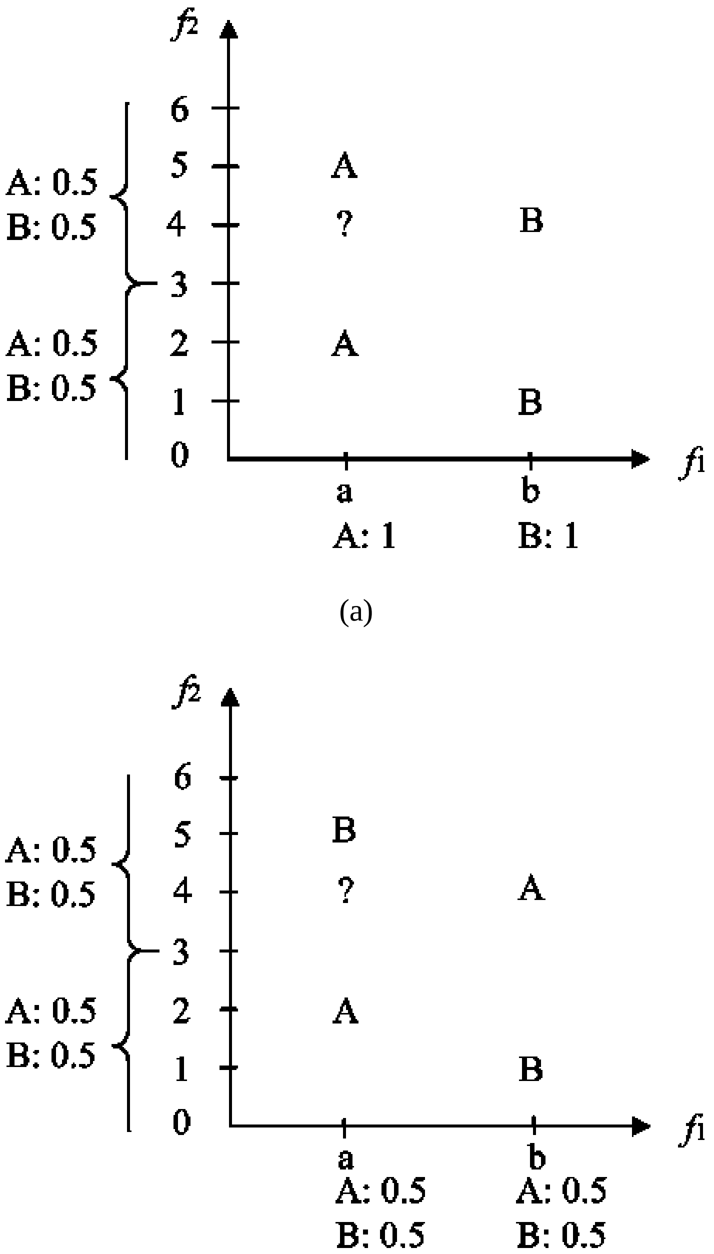
On the other hand, the rules for f_2 are:

If f_2 = -∞..3 Then vote[A]=0.5, vote[B]=0.5.
If f_2 = 3..∞ Then vote[A]=0.5, vote[B]=0.5.

For the query instance marked as "?" in Fig. 1, feature f_1 casts its vote only for class A. On the other hand, f_2 casts half of its vote for class A, and the other half for B. In total, class A gets 1.5 votes, while class B receives only 0.5 votes. Since the class A receives more votes than B, the class of the query instance is predicted as A.

Note that the feature f_2 is irrelevant in this simple data set. The rules learned for that feature will distribute their votes equally among both classes, and therefore they will not have an effect on the outcome of classification. This shows that the VFC algorithms are robust to irrelevant features (Güvenir, 1998).

Since the VFC algorithms, as introduced above, learn rules that contain a single condition based on a single feature in their antecedent, they fail in domains where antecedents of the rules must contain conditions involving two or more features. A simple such data set is shown in Fig. 1b. In this case, the rules for both features will distribute their votes equally among both classes, and the classifier will have to make a random guess.



(a)

(b)

Fig. 1. Learning a model and classification by VFCC; a) a suitable data set, b) a problematic data set.

The Voting Features based Classifier with feature Construction (VFCC) algorithm introduced in this paper uses a feature construction technique in order to cope with such cases. The feature constructor of VFCC forms composite features, from features that are not *decisive*. We say that a feature is decisive if the difference between the maximum and minimum votes of the rules on that feature is high. Given two features f' and f'', the possible values of the composite feature, represented as f'& f'', are pairs in the form (v' and v''), where v' is one of values of f' and v'' is one of values of f''. After the feature construction step, the VFCC will learn the following rules:

If f_1& f_2 = (a & 3..∞) Then vote[A]=0, vote[B]=1.0.
If f_1& f_2 = (b & -∞..3) Then vote[A]=0, vote[B]=1.0.
If f_1& f_2 = (a & -∞..3) Then vote[A]=1.0, vote[B]=0.
If f_1& f_2 = (b & 3..∞) Then vote[A]=1.0, vote[B]=0.

With these new rules, the VFCC will predict the class of the query instance in Fig. 1b as B. This example shows that a decisive feature can be constructed from two indecisive ones. Therefore indecisive features are potentially good candidates for constructing decisive features.

In this paper, we also show the application of the VFCC to the problem of predicting if a bank will encounter financial distress, by analyzing some ratios derived directly from its current financial statements. The VFCC algorithm proposed in this work constructs rules whose antecedents may contain conjuncts based on several features. Experimental results on recent financial statements of banks in Turkey show that the VFCC algorithm performs better than other well-known classification algorithms.

One of the earliest attempts in feature construction was the BACON system (Bradshaw, Langley, & Simon, 1980). It is a program that discovers relationships among real-valued features of instances in data, and uses two operators, namely, multiplication and division. Utgoff described the feature construction problem and investigated overlapping feature construction methods for game playing (Utgoff, 2001). Kim and Choi proposed a discriminant analysis method, called C-LDA, using composite features for the pattern classification problem (Kim, & Choi, 2007). Their composite feature concept is motivated from a windowed feature in an image, which consists of a number of pixels. Piramuthu used feature construction for reduction of tabular knowledge-based systems (Piramuthu, 2004). Hanczar et al. proposed a feature construction technique based on synergic interactions between gene pairs (Hanczar, Zucker, Henegar, & Saitta, 2007).

The next section describes the VFCC algorithm in detail. Section 3 introduces the problem of predicting financial distress risks of a bank given its financial ratios. Section 4 explains the data set that was used in predicting the risk of financial distress using the VFCC algorithm. Section 5 presents the results of our experiments using the VFCC and the other well-known classification algorithms implemented in the Weka package (Witten, & Frank, 2005). Finally, the last section concludes with some remarks and suggestions for future work.

2 Voting Features based Classifiers with Feature Construction

The VFCC algorithm is shown in Fig. 2. The details of the training, feature construction and classification algorithms are explained in the following sections.

2.1 Training

In its first step, the training algorithm converts each continuous feature into a categorical one. In order to do that, for each class, the median of the feature values of all training instances is found. Let m_c be the median of all training instances for class c, and C be the number of classes. Then, these medians are sorted in increasing order. Let the ordered list of medians be m_1, m_2, … m_{C}. The categorical values for that feature are

$$\left(-\infty .. \frac{m_1+m_2}{2}\right),\ \left(\frac{m_1+m_2}{2} .. \frac{m_2+m_3}{2}\right),\ \ldots,\ \left(\frac{m_{C-1}+m_C}{2} .. \infty\right).$$

That is, each new categorical value represents a range of continuous values for that feature. Note that the number of categorical values is equal to C. For each instance, the continuous value of that feature is then replaced by the new categorical value representing the range that covers the continuous value. This way of determining cut-off points guarantees that the accuracy of each such feature, after categorization, is at least the default accuracy.

```
train (trainingSet)
    for each feature f
        if f is continuous, makeCategorical (f)
        for each categorical value v of f
            for each class c
                $vote_{f,v}[c]$ = P(f has valuev | c) using instances in trainingSet
end // train

constructFeatures (F) // F: Set of primitive features
    initialize candidateFeatures = empty
    initialize goodFeatures = empty
    for each feature f in F
        if maxVoteDiff[f] > $\tau$
            add f into goodFeatures
        else
            add f into candidateFeatures
    constructedFeatures = makeFeaturesFromAllPairsOf(candidateFeatures)
    while candidateFeatures is not empty
        constructedFeatures = makeFeaturesFromAllPairsOf(candidateFeatures)
        sort constructedFeatures in descending order of maxVoteDiff[f]
        for each feature f in constructedFeatures
            if maxVoteDiff[f] > $\tau$ and parents of f are in candiateFeatures
                add f into goodFeatures
                remove the parents of f from candidateFeatures
        Let f be the constructedFeatures with minimum maxVoteDiff[f] and
            parents of f are in candiateFeatures
        add f into candidateFeatures
        remove the parents of f from candidateFeatures
end // constructFeatures

classify (q)
    for each class c, totalVote[c]=0
    for each feature f
        if $q_f$ value is known
            for each class c
                totalVote[c] = $vote_{f,v}[c]$
    return $\arg\max_c$ (totalVote[c])
end // classify
```

Fig. 2. The VFCC Algorithm.

The model constructed by the training algorithm is composed of vote values for each class, given a feature and value pair. The vote$_{f,v}$[c] is defined as the probability that an instance of class c, in the training set, has the value v for feature f. Since the votes are defined as probabilities, $\sum_{c=1}^{C} \text{vote}_{f,v}[c] = 1$, that is, given a value v, a feature f distributes its vote among the classes.

2.2 Constructing new Features

The constructFeatures algorithm, the heart of VFCC, constructs new features from pairs of known features. The VFCC algorithm first runs the training algorithm using the primitive (given) feature set. The construcFeatures algorithm first initializes two lists; candidateFeatures and goodFeatures. Among the primitive features, the decisive ones are put into the goodFeatures list. For a given feature-value pair, the vote difference, VD, is the difference between the maximum and minimum votes. For a given feature, among all its possible values, the maximum of these values is called maxVD. We say that a feature is decisive if its maxVD is more than a given threshold. All decisive features can be used in classification. On the other hand, indecisive features are candidates for constructing decisive new features, and they are put into the list candidateFeatures.

From all pairs of features in candidateFeatures, new features are constructed, and put into a new list called constructedFeatures. Given two features f_i with possible values V_i and f_j with possible values V_j, a new feature f_i & f_j is constructed whose possible values are the Cartesian product of $V_i \times V_j$, that is $\{(v,w) \mid v \in V_i \text{ and } w \in V_j\}$. Once such a new feature is constructed, the values of this feature are computed for all training instances, and the votes are computed. The newly constructed features in candidateFeatures are sorted in a decreasing order of decisiveness. If the first one is decisive (its maxVoteDiff is more than threshold τ), it is placed into the goodFeatures list. In order to guarantee the independence among the features to be used in classification, the features that were used in the construction of the new good feature, called the parents, must be removed from the candidateFeatures. In other words, a primitive feature f_i and constructed feature f_i & f_j must not both be used at the same time in the classification. Then, in the same order, the other decisive features in the constructedFeatures list are also put into the goodFeatures list as long as their parents are still in the candidateFeatures list.

Using the heuristic that indecisive features are good candidates for constructing decisive features, in its last step, the constructFeatures algorithm adds the least decisive feature from the constructedFeatures list, whose parents are still in the candidateFeatures lits, into the candidateFeatures list. It also removes its parents from the candidateFeatures list to guarantee independence.

After completing the feature construction step, the VFCC algorithm is ready to classify the query instances using the set of features in the goodFeatures list.

2.3 Classification

For a given query instance *q*, the classifier collects the votes of each feature. If the value of *q* for a feature *f*, that is q_f, is unknown, that feature does not participate in the voting. After collecting the votes of each feature, the classifier declares the class label of *q* as the class that received the maximum amount of votes.

3 Financial Distress Analysis

It has been observed over the past 30 years that, despite the presence of more sophisticated markets and well established banking systems, there have been significant bank failures and bank crises, especially recently. A well-organized and efficient banking system is an essential prerequisite for economic stability and growth of a country. Banks play an important role in the functioning of an organized money market. They act as a conduit for mobilizing funds and channelizing them for productive purposes. Because of its central position in the economy, the banking sector is one of the most strictly regulated sectors in modern economies (Fukuda, Kasuya, & Akashi, 2008). This is especially important in transition economies since the health of the banking sector is a prerequisite to increase private savings and allocate loans to their most productive use (Lanine, & Vennet, 2006). Central bankers fear widespread bank failures because they exacerbate cyclical recessions and may trigger a financial crisis (Westernhagen, et al., 2004). Bank failures pose a direct threat to the economy of any country, even to the global economy, and hence regulatory changes are required in order to decrease the risks and reduce their costs. Bank failures are usually followed by unfavorable consequences on stakeholders outside the failed banks themselves. Sometimes the consequences are felt by the non-banking systems as well. A failure can result in much harm to employment, earnings, financial development and other associated public interests (Apea, & Sezibera, 2002). To prevent systemic banking crisis, bank regulators are interested in developing early warning systems (EWS) in order to identify problem banks and avoid bankruptcies (Tung, Quek, & Cheng, 2004; Lanine, & Vennet, 2006; Ng, Quek, & Jiang, 2008).

Financial distress, as a dynamic and mostly lengthy process, starts with the deterioration of the financial structure of a healthy economic agent below a threshold level (considered normal-healthy) -which usually cannot be determined- due to an abrupt and short-lived event or a chain of events or due to repeated anomalies occurring for a long period of time. The significance of the financial distress for the firm and the whole economy itself, though, would matter much more than the process itself, because, the temporariness or the permanence and the length of the period of distress would determine the viability of the firm in the long run. This is significant, as, if one sufficiently big agent encounters the distress the whole economy may be influenced by this particular event. The same holds for a large group of small firms that are members of a particular industry especially if the industry is heavily vertically and/or horizontally integrated.

As to banks, sharing the largest portion of the assets of, and operating in many different areas of the financial industry, as they are the biggest suppliers of funds to the real sector, financial distress of especially a large one or several may result in the collapse of the whole banking and finance sector, and the whole economy per se. Hence, the prediction of financial distress of the individual banks and the banking sector is of utmost importance, for the authorities, monitoring bodies and even for the banks themselves.

National regulatory authorities collect information from banks about their financial state in the form of quarterly balance sheets. They derive many ratios from these absolute quantities. Using these ratios, the authorities try to foresee a possible financial distress that a bank may encounter. They would like to know which of these ratios and what values of these ratios can be used to predict a possible financial distress in following few quarters, so that they can take corrective actions if necessary. Along with high classification accuracy, the learned model has to be verifiable by human experts. The following section summarizes a dataset compiled for such purposes.

4 The Data Set

The dataset used in this study is formed by using quarterly financial reports of 46 Turkish banks, gathered from the official web site of The Banks Association of Turkey. The quarterly periods start from December 2002 and go until March 2007, involving 18 periods. The dataset comprises 59 predictive features (all continuous) and one class attribute. The features and their descriptions are listed in Table 1. The feature values are composed of financial ratios that are originally computed by the banks. These feature values can be summarized in eight different categories: Assets Quality ratios, Asset Quality Index ratios, Balance-Sheet Structure ratios, Capital Adequacy Ratios, CAPital ratios, Income-Expenditure structure ratios, Liability Structure ratios, LIQuidity ratios, and PRofitability ratios. All the ratios are calculated at period *t*, by using Turkish Lira denominated financial reports. Assuming that economic policies and economy wide changes are almost perfectly reflected in bank financial reports, macroeconomic and other factors are not taken into consideration.

Each instance in the data set represents the ratios derived from the balance sheet of a bank that was profitable at a quarter *t*. Here *t* represents 15 different quarters in the range 2002 Q4 to 2006 Q2. The class attribute has two values, namely Success and Failure. The class attribute at period *t*, is determined by using profit values of the following three periods, as shown in Fig. 3. An instance representing a bank that is profitable at quarter *t* and also in the following three quarters, *t*+1, *t*+2 and *t*+3, is labeled as Success at that period *t*. On the other hand, an instance representing a profitable bank at quarter *t* is labeled as Failure if it either incurred losses at all the following periods *t*+1, *t*+2, and *t*+3 or made profits at period *t*+2 but incurred losses at periods *t*+1 and *t*+3. The other cases are excluded from the dataset.

Table 1. Features and their descriptions.

Feature	Description	Feature	Description
AQ_1	Financial Assets (Net) / Total Assets	AQ_5	Loans Under Follow-Up (Net) / Total Loans
AQ_2	Total Loans / Total Assets	AQ_6	Specific Provisions / Loans Under Follow-Up
AQ_3	Total Loans / Total Deposits	AQ_7	Permanent Assets / Total Assets
AQ_4	Loans Under Follow-Up (Gross) / Total Loans	AQ_8	Consumer Loans / Total Loans
AQI_1	Past Due Loans (Net) / Average Total Assets		
AQI_2	Subsidiaries And Associated Companies (Net) + Fixed Assets (Net) / Average Total Assets		
AQI_3	Past Due Loans (Net) / Total Loans		
AQI_4	Provisions For Past Due Loans / Average Total Loans		
BSS_1	Tc Assets / Total Assets	BSS_5	Tc Loans / Total Loans
BSS_2	Tc Liabilities / Total Liabilities	BSS_6	Total Deposits / Total Assets
BSS_3	Fc Assets / Fc Liabilities	BSS_7	Funds Borrowed / Total Assets
BSS_4	Tc Deposits / Total Deposits		
CAR_1	Shareholders' Equity / (Amount Subject To Credit + Market + Operational Risk)		
CAR_2	Shareholders' Equity / Total Assets		
CAR_3	(Shareholders' Equity - Permanent Assets) / Total Assets		
CAR_4	Net On Balance Sheet Position / Total Shareholders' Equity		
CAR_5	Net On And Off Balance Sheet Position / Total Shareholders' Equity		
CAP_1	Shareholders' Equity / Average Total Assets	CAP_5	Loans Under Follow-Up (Net) / Shareholders' Equity
CAP_2	Liabilities / Shareholders' Equity	CAP_6	Total Loans (Net) / Shareholders' Equity
CAP_3	Paid Up Capital / Shareholders' Equity	CAP_7	Subsidiaries And Associated Companies (Net) / Shareholders' Equity
CAP_4	Free Capital / Shareholders' Equity		
IE_1	Net Interest Income After Specific Provisions / Total Assets		
IE_2	Net Interest Income After Specific Provisions / Total Operating Income		
IE_3	Non-Interest Income (Net) / Total Assets		
IE_4	Other Operating Expenses / Total Assets		
IE_5	Personnel Expenses / Other Operating Expenses		
IE_6	Non-Interest Income (Net) / Other Operating Expenses		
LS_1	Total Loans / Deposits	LS_2	Deposits / Liabilities
LIQ_1	Liquid Assets / Total Assets		
LIQ_2	Liquid Assets / Short-Term Liabilities		
LIQ_3	Tc Liquid Assets / Total Assets		
LIQ_4	Cash And Dues From Central Bank, Other Banks And Money Market / Demand + Term Deposits		
LIQ_5	Liquid And Quasi-Liquid Assets / Average Total Assets		
PR_1	Net Profit/Losses / Total Assets	PR_5	Total Expenses / Average Total Assets
PR_2	Net Profit/Losses / Total Shareholders' Equity	PR_6	Net Of Interest Income / Average Total Assets
PR_3	Income Before Taxes / Total Assets	PR_7	Net Of Interest Expense / Average Total Assets
PR_4	Total Income / Average Total Assets	PR_8	Non-Interest Expenses / Average Total Assets
PR_9	Profit (Loss) For The Period / Average Shareholders' Equity		
PR_10	Interest Income On Loans - Interest Paid For Deposits / Net Of Interest Income (Interest Expense)		
PR_11	Total Income / Total Expenses		
PR_12	Total Interest Income / Total Interest Expenses		
PR_13	Non-Interest Income / Non-Interest Expenses		
PR_14	Interest Income / Total Income		
PR_15	Interest Expenses / Total Expenses		
Class	Success or Failure		

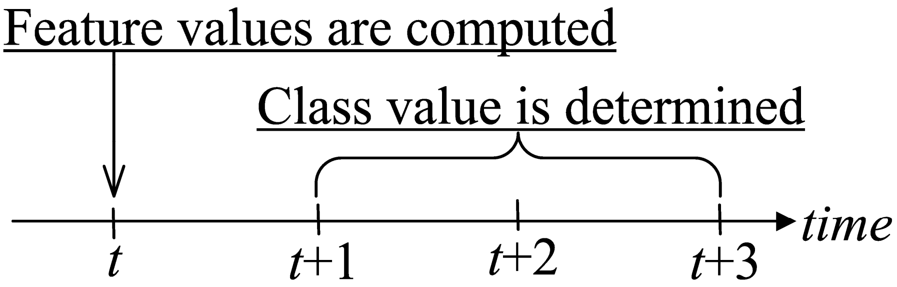


Fig. 3. Periods over which the feature and class values are determined.

The data set contains 690 instances; 607 of them are labeled as “Success” and 83 as “Failure.” There are 2343 (5.7%) missing feature values.

5 Experimental Results

The VFCC algorithm has been implemented in the Java language and compared with all other rule learning classifiers available in the Weka package (Witten, & Frank, 2005). Accuracy values attained through stratified 10-fold cross-validation results are shown in Table 2. Results of some other classifiers are also included in the table for comparison.

We have also investigated the effect of the choice of the threshold on the accuracy of VFCC. As seen in Fig. 4, higher values of threshold τ result in slightly higher values of accuracy, up to a certain point. High values of τ result in a smaller number of more decisive rules, while low values result in a greater number of rules, including some less decisive rules along with the more decisive ones. Since the low quality rules have low effect in the voting step of classification, the accuracy is determined by the decisive rules. High threshold values also cause more pairs of features to be tested during the construction process. The rules learned with high τ values will include many conjuncts in their antecedents, which are very accurate but difficult to interpret by human experts, that is they overfit the training set. Such rules can be ignored in applications such as knowledge acquisition.

Table 2. 10-fold cross-validation comparison results. The VFCC results are shown in bold face.

Classifier	**Accuracy (%)**
Voting Features Classifier with Feature Construction (VFCC)	**90.72**
RIpple DOwn Rule Learner(Ridor)	90.00
NNGE classifier (non-nested generalized exemplars)	90.00
PART	89.71
REPTree	89.57
OneR	89.42
J48 pruned tree	88.55
Alternating decision tree (ADTree)	88.41
SMO for training SVM using polynomial kernels	88.26
Single conjunctive rule learner:	87.97
ZeroR	87.97
Voting feature intervals (VFI) classifier	87.83
RandomTree	87.83
Decision Stump	87.68
Voted Perceptron	87.25
Instance Based IB1 classifier	85.07
Naive Bayes Classifier	84.64

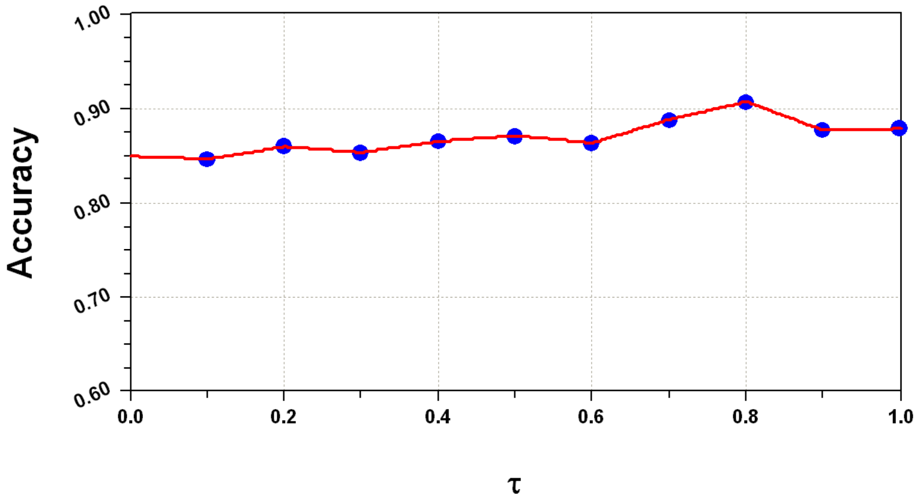


Fig. 4. The effect of τ on accuracy.

In our experiments with the dataset mentioned above, although the effect of the choice of τ in the accuracy is low, we found that 0.8 is the optimum value for our dataset. Using all instances in the training, the VFCC algorithm has learned 30 rules, for $\tau = 0.8$. Some of the rules learned are shown in Fig. 5. All the rules that match a given query instance are used in the voting. The model learned by the VFCC algorithm is a set of simple rules. There is no ordering imposed on the rule set learned. Therefore, each of the rules constructed by the VFCC algorithm can easily be verified individually by human experts. For example, the rule

```
If BSS_5&PR_3="0.6898..∞&-∞..-0.0017"
Then SUCCESS (V=0.0582) FAILURE (V=0.9418) VD=0.8836 Sp=45
```

is interpreted as if BSS_5 (Tc Loans / Total Loans) is more than about 0.7 and PR_3 (Income Before Taxes / Total Assets) is less than about -0.002 than the bank will face distress in the next three periods with about %95 certainty. Here, Tc refers to the loans received in Turkish currency, while Total Loans refers to the Turkish currency equivalent of all loans received.

6 Conclusion

In this paper, a new method for constructing new features from initially given (primitive) features is proposed. The VFCC algorithm is an extension to the VFC algorithms that learn rules that are based on only one feature. In domains where rules involve conditions on two or more features, the VFC algorithms fail. The feature construction algorithm of VFCC employs a heuristic that good (decisive) rules can be constructed by combining indecisive ones. The VFCC algorithm has been applied to the problem of predicting bank financial distress, by analyzing and comparing current and previous financial ratios of banks in Turkey derived from their financial statements. Experimental results show that the VFCC algorithm achieves better accuracy than all other rule learning classification algorithms, implemented in the Weka package. Another important advantage is that, the rules learned by the VFCC algorithm can be easily evaluated and verified by human experts.

```
If PR_11&AQ_8="1.0704..∞& 0.0247..∞"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=87
If PR_12&AQ_3="0.579..∞&-∞.. 0.546"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1 Sp=41
If CAP_1&CAP_2="0.4444..∞& 1.2383..∞"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=5
If BSS_6&BSS_7="0.2656..∞& 0.1055..∞"
Then SUCCESS (V=0.9598) FAILURE (V=0.0401) VD=0.9197 Sp=176
If PR_4&IE_5="-∞..0.1377&-∞..0.4269"
Then SUCCESS (V=0.0436) FAILURE (V=0.9564) VD=0.9128 Sp=4
If AQI_1&AQ_4="0.0195..∞&-∞..0.0694"
Then SUCCESS (V=0.9535) FAILURE (V=0.0465) VD=0.907 Sp=151
If BSS_5&PR_3="0.6898..∞&-∞.. -0.0017"
Then SUCCESS (V=0.0582) FAILURE (V=0.9418) VD=0.8836 Sp=45
If AQ_5&AQ_6="-∞..0.0016&-∞..0.9364"
Then SUCCESS (V=0.0758) FAILURE (V=0.9242) VD=0.8484 Sp=8
If CAP_3&PR_1="0.5291..∞&-∞..-0.0036"
Then SUCCESS (V=0.08) FAILURE (V=0.92) VD=0.84 Sp=72
If CAP_6&LS_1="0.5881..∞&-∞..0.8038"
Then SUCCESS (V=0.9111) FAILURE (V=0.0889) VD=0.8222 Sp=152
If PR_8&PR_9="0.0818..∞&-∞..-0.0151"
Then SUCCESS (V=0.089) FAILURE (V=0.911) VD=0.822 Sp=60
If CAP_7&LIQ_4="0.0175..∞&-∞..0.369"
Then SUCCESS (V=0.9086) FAILURE (V=0.0914) VD=0.8172 Sp=221
If PR_14&IE_6="0.7733..∞&0.5054..∞"
Then SUCCESS (V=0.9078) FAILURE (V=0.0922) VD=0.8156 Sp=146
If LIQ_5&PR_2="-∞..1.3887&-0.0153..∞"
Then SUCCESS (V=0.9072) FAILURE (V=0.0928) VD=0.8144 Sp=290
If CAR_2&CAR_3="0.238..∞&-∞..0.0943"
Then SUCCESS (V=0.0961) FAILURE (V=0.9039) VD=0.8078 Sp=16
If IE_1&PR_5&IE_2="0.0238..∞&-∞..0.1169&-∞..0.5657"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=23
If IE_1&PR_5&IE_2="-∞..0.0238&0.1169..∞&0.5657..∞"
Then SUCCESS (V=0.0835) FAILURE (V=0.9165) VD=0.833 Sp=5
If AQ_2&LS_2&CAR_1="0.2566..∞&-∞..0.4737&-∞..0.3618"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=50
If PR_15&CAR_5&BSS_2="0.3659..∞&0.0003..∞&0.5803..∞"
Then SUCCESS (V=0.9089) FAILURE (V=0.0911) VD=0.81788 Sp=74
If LIQ_1&AQI_2&AQI_3="-∞..0.4223&-∞..0.1021&0.1251.. ∞"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=46
If IE_4&CAP_4&AQI_4="-∞..0.0357&-∞..0.5378&-∞..0.0886"
Then SUCCESS (V=1.0) FAILURE (V=0.0) VD=1.0 Sp=45
```

Fig. 5. Rules learned using all instances in training, τ=0.8. Here, V: vote, VD: Vote Difference, Sp: Support.

The VFCC algorithm uses a threshold τ that takes on a value between 0 and 1. In our experiments, we tried 10 values with 0.1 increments. It has been observed that the choice of τ has a minimal effect on the accuracy. However it affects the number and quality of the rules constructed.

The quality of the model learned by the classifier depends, among other factors, on the training set. We plan to extend the dataset with more instances in the future. With more instances, the VFCC algorithm is expected to find better boundary values when converting continuous features to nominal ones.

We plan to develop an early warning system that monitors the quarterly financial statements of the banks in Turkey and alerts the experts about the banks that should be further investigated. The knowledge base of the system will be updated at the end of each quarter with the new set of statements provided.

Acknowledgements

This work was supported in part by the Scientific and Technical Research Council of Turkey, under Grant EEEAG-105E065. The authors thank Narin Emeksiz of Central Bank of the Republic of Turkish for her invaluable comments in the manuscript.

References

Apea, C., & Sezibera, J. (2002). Some Causes of Bank Failure; A Case Study of Ghana Co-operative Bank Ltd., International Accounting and Finance Master Thesis No 2002:54, Göteborg University, Graduate Business School, School of Economics and Commercial Law, ISSN 1403-841X.

Asuncion, A. & Newman, D.J. (2007). UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

Bradshaw, G. L., Langley, P., & Simon, H. A: (1980). BACON.4: The discovery of intrinsic properties. In Proceedings of the Third National Conference of the Canadian Society for Computational Studies of Intelligence (1980) 19-25.

Fukuda S., Kasuya, M., & Akashi, K. (2008). Impaired bank health and default risk, *Pacific-Basin Finance Journal* (doi:10.1016/j.pacfin.2008.06.002)

Güvenir, H. A. Demiröz, G., & Ilter, N. (1998). Learning Differential Diagnosis of Erythemato-Squamous Diseases using Voting Feature Intervals, *Artificial Intelligence in Medicine*, *13*(3) (1998) 147-165.

Güvenir, H. A., & Sirin I. (1996). Classification by Feature Partitioning, *Machine Learning*, *23*(1), 47-67.

Güvenir, H. A., Emeksiz, N. Ikizler, N., & Örmeci, N. (2004). Diagnosis of gastric carcinoma by classification on feature projections, *Artificial Intelligence in Medicine*, *31*(3), 231-240.

Güvenir, H. A., (1998). A Classification Learning Algorithm Robust to Irrelevant Features, in Artificial Intelligence: Methodology, Systems Applications (Proceedings of AIMSA'98), Fausto Giunchiglia (Ed.), Springer-Verlag, *LNAI 1480*, Sozopol, Bulgaria, 281-290.

Hanczar, B., Zucker, J-D., Henegar, C., & Saitta, L. (2007). Feature construction from synergic pairs to improve microarray-based classification, *Bioinformatics*, *23* (21). 2866–2872.

Kim, C., & Choi, C. H. (2007). A discriminant analysis using composite features for classification problems, *Pattern Recognition*, *40*(11). 2958-2966.

Lanine, G., & Vennet, R. V. (2006). Failure prediction in the Russian bank sector with logit and trait recognition models, *Expert Systems with Applications*, *30*(3). 463-478.

Ng, G.S., Quek, C., & Jiang, H. (2008). FCMAC-EWS: A bank failure early warning system based on a novel localized pattern learning and semantically associative fuzzy neural network, *Expert Systems with Applications*, *34*(2). 989-1003.

Piramuthu, S. (2004). Feature construction for reduction of tabular knowledge-based systems, *Information Sciences*, *168*(1-4). 201-215.

Tung, W.L., Quek, C., & Cheng, P. (2004). GenSo-EWS: a novel neural-fuzzy based early warning system for predicting bank failures, *Neural Networks*, *17*(4). 567-587.

Utgoff, P.E. (2001). Feature Construction for Game Playing, in Fuerenkranz & Kubat (Eds.), Machines that Learn to Play Games. Nova Science Publishers. 131-152.

Westernhagen, N., Harada, E., Nagata, T., Vale, B., et.al., (2004). Bank Failures in Mature Economies, Basel Committee on Banking Supervision Working Paper No. 13.

Witten, I. H., & Frank, E. (2005) Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco.