# An Ensemble Approach for a Cross-selling Business Problem

The YNUDM Team*

School of Information Science and Engineering, Yunnan University, Kunming, 650091, China

**Abstract.** In this paper, we analyze the difficulties of the competition task and propose the thought of ensemble approach which includes two processes: data preprocessing and mining prediction. In data preprocessing, redundant attributes, dirty data, null value and data sampling are removed and processed effectively. In mining prediction, we use decision tree, artificial neural network and their ensemble approach as mining tools. We evaluate these methods by ROC curve and find which the better solution to this problem is. In the end of this paper, the business insights interpreted from our scoring model results are also presented.

## 1 Introduction

The PAKDD'07 Data Mining Competition task is described as following:

"The company currently has a customer base of credit card customers as well as a customer base of home loan (mortgage) customers. Both of these products have been on the market for many years, although for some reason the overlap between these two customer bases is currently very small. The company would like to make use of this opportunity to cross-sell home loans to its credit card customers, but the small size of the overlap presents a challenge when trying to develop an effective scoring model to predict potential cross-sell take-ups…"

The report will disclose our solution to this task. The rest of the report is organized as follows. We discuss the difficulties of the task in Section 2. In Section 3, the process of data mining prediction of potential cross-sell take-ups is analyzed. The data preprocessing process is introduced in Section 4. Section 5 describes our algorithms. This is a thorough ensemble approach, for it is a combination approach of decision tree and ANN which is based on combining decision tree classifiers and ANN classifiers separately. Analysis and experimental evaluations are presented in Section 6 and we give our conclusions in Section 7.

## 2 Analysis

After analyzing the task carefully, we have found the following difficulties we have to face.

- In the data set, the number of credit card customers is almost 60 times of that of home loan customers, which makes a class-imbalance setting. Then how can we deal with such class-imbalance?
- From the potential interest of the company, it may be anticipated that no potential home loan user is missed. This implies that missing a loan user is with a bigger cost than mistakenly classified a credit card user as a loan user. Then, how can we deal with such different misclassification costs?

* The team comprises Yuebo Li, Yu Su, Lili Feng and Lizhen Wang. Correspondence: Lizhen Wang, Tel.: +86-871-503-4096, Email: lzhwang@ynu.edu.cn.

- The data set, which is used to model and predict, has 40 modeling variables. Among these 40 variables, some of them contribute to the target variable, some of them are redundant, and some of them are useless for prediction. Then how can we recognize and choose them?
- In modeling and prediction datasets, there are large number of null values and some dirty data. These data will effect the results of predicting in despite of what methods are used. Then how can we deal with the null values and dirty data in our approach?

At present there are many techniques which could address the above difficulties separately, such as techniques for learning with class-imbalance learning, cost-sensitive learning, and semi-supervised learning. However, tackling these difficulties simultaneously is still an interesting challenge. The next section will present an ensemble approach. It is anticipated that such an approach can also be useful in other scenarios.

## 3 The Process of Prediction of the Ensemble Approach

The process of the ensemble approach is shown as Fig. 1. There are two processes. They are the data preprocessing and the mining prediction. The data preprocessing process converts the original data into data for prediction, in which the primary jobs are dealing with null and dirty values, redundant and useless attributes eliminating, and training datasets choosing. The mining prediction performs the prediction of potential cross-sell take-ups. For finding better solutions, this step takes an approach of classifier combination. Based on a hybrid sampling method (hybrid of the undersampling method and the oversampling method), some decision tree classifiers and ANN classifiers (called base classifier) are built. And then we predict potential cross-sell take-ups by voting the prediction of each base classifier.

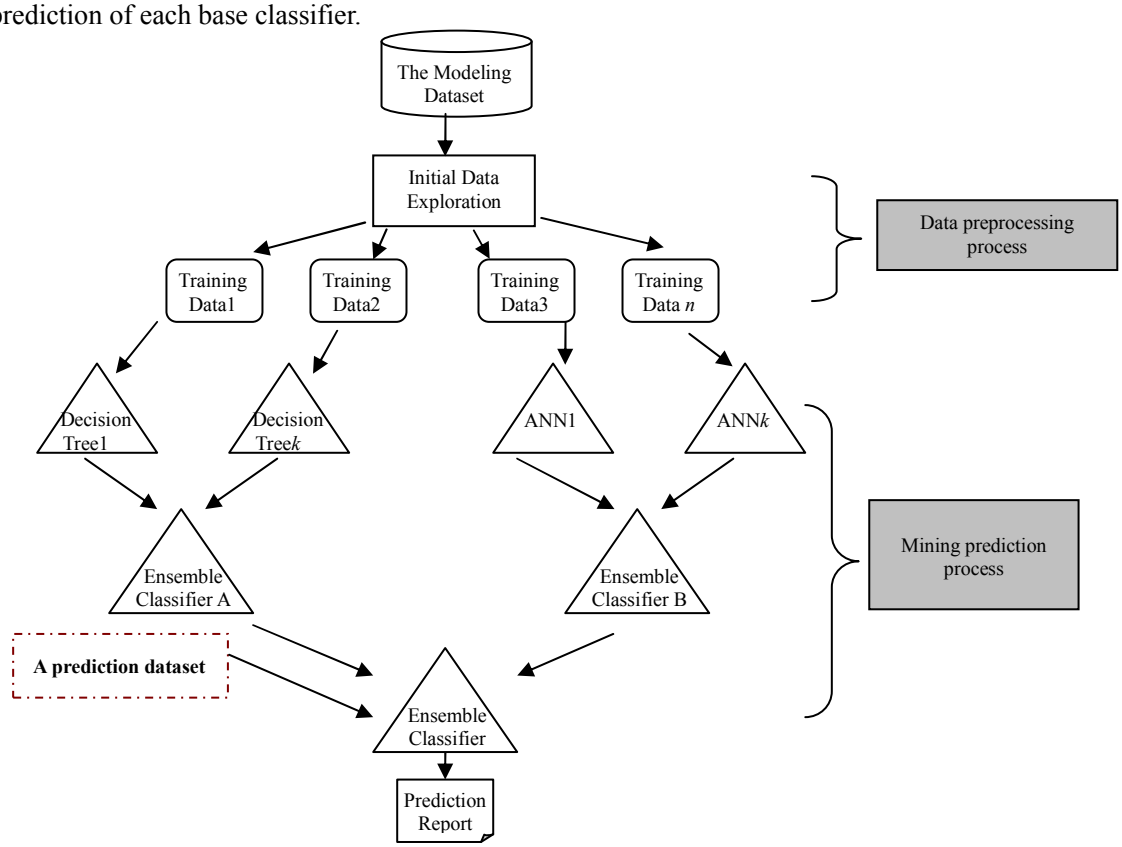


**Fig. 1 The process of data mining prediction of potential cross-sell take-ups**

## 4 The data preprocessing process

The data in the real-world datasets provided by the consumer finance company includes relevant attributes for mining purpose. It's highly susceptible to noisy, dirty, missing, and inconsistent data. There is a need to pre-process these redundant and useless data to make it easier to mine for our task.

### 4.1 Redundant and useless attributes processing

We use an analysis tool based on information entropy for each attribute to produce a score indicating the correlation between the attribute and TARGET_FLAG. The higher the score is, the closer the correlation. The result is listed in Table 1. (In descending order):

Table.1. Scores of Attributes indicating the correlation between the attribute and TARGET_FLAG

| Attribute | Score | Attribute | Score | Attribute | Score |
|---|---|---|---|---|---|
| AGE_AT_APPLICATION | 0.077 | B_ENQ_L12M_GR1 | 0.025 | A_TOTAL_NBR_ACCT | 0.007 |
| CUSTOMER_SEGMENT | 0.070 | B_ENQ_L3M | 0.024 | A_TOTAL_BALANCES | 0.006 |
| CURR_RES_MTHS | 0.066 | DISP_INCOME_CODE | 0.022 | VISA_CARD | 0.006 |
| B_ENQ_L12M_GR3 | 0.065 | PREV_EMPL_MTHS | 0.020 | NBR_OF_DEPENDANTS | 0.005 |
| B_ENQ_L6M_GR3 | 0.051 | B_ENQ_LAST_WEEK | 0.019 | CREDIT_CARD_TYPE | 0.003 |
| PREV_RES_MTHS | 0.039 | B_ENQ_L6M_GR1 | 0.019 | DVR_LIC | 0.003 |
| B_ENQ_L6M | 0.032 | MARITAL_STATUS | 0.017 | RETALL_CARDS | 0.003 |
| CURR_EMPL_MTHS | 0.030 | B_DEF_PAID_IND | 0.016 | AMEX_CARD | 0.003 |
| OCCN_CODE | 0.030 | B_DEF_PAID_L12M | 0.016 | MASTERCARD | 0.002 |
| B_ENQ_L1M | 0.026 | B_DEF_UNPD_IND | 0.016 | CHQ_ACCT_IND | 0.001 |
| B_ENQ_L12M_GR2 | 0.026 | ANNUAL_INCOME_RANGE | 0.016 | DINERS_CARD | 0.001 |
| TOTAL_NBR_CREDIT_CARDS | 0.026 | A_DISTRIC_APPLICANT | 0.011 | B_DEF_UNPD_L12M | 0.000 |
| TOTAL_AMT_DELQ | 0.025 | SAV_ACCT_IND | 0.008 | | |
| B_ENQ_L6M_GR2 | 0.025 | RENT_BUY_CODE | 0.008 | | |

According to Table 1, we remove the attributes whose score <0.005.

There are some attributes whose score is not less than the given threshold needing removing with other criterion. Take ANNUAL_INCOME_RANGE and DISP_INCOME_CODE for example. The score of both of them is more than 0.005. But we can remove one of them according to the obvious close relationship of these two attributes. Notice that there are many null values in DISP_INCOME_CODE, so it's removed.

### 4.2 Dirty data and null value processing

Some obvious dirty data have been discovered before learning the training set. We list them as follows:

1) There is 16 values with "1000" (83 years) in CURR_EMPL_MTHS. We consider them as a special class;

2) There is miswritten value in ANNUAL_INCOME_RANGE;

3) There are 5 missing values in ANNUAL_INCOME_RANGE. We consider them as a special class.

4) There is a *M* value in SAV_ACCT_IND inconsistent with the introduction in data

dictionary. We change *M* into *N*;

5) There are some values in A_DISTRICT_APPLICANT inconsistent with the introduction in data dictionary.

As to null values, we adopt C4.5 to process them dynamically. If the null value is numerical, we replace it with the average of other non-null values; else we replace it with the value taking the mode.

### 4.3 Sampling processing

There are 700 records with TARGET_FLAG=1 and 40,000 records with TARGET_FLAG=0 in the modeling dataset of 40,700 customers. It's a typical unbalancing class problem. So we adopt oversampling and undersampling to sample.

Oversampling replicates positives till the amount of positives is as same as that of negatives. It will be applied for the set of tuples whose TARGET_FLAG is 1. Here, we replicate negatives for 5 times.

Undersampling is aimed at negatives sampling. It's required that original records should be classified with certain criterion (e.g. sorting). Then start sampling randomly in these classes. It will be applied for the set of tuples whose TARGET_FLAG is 0. Here, we sorted attributes with highest score in table 1 (attributes AGE_AT_APPLICATION and B_ENQ_L6M_GR3) to classify the original records, and 23 classes were obtained. In each class, we select records at a ratio about 50% randomly to form training set.

## 5 The mining prediction process

We use an ensemble method of the decision tree and the artificial neural network as mining tools.

### 5.1 Decision tree

#### 5.1.1 Motivation and approach

To get the class label on prediction dataset, we adopt decision tree to classify the modeling dataset mainly based on the following characteristics: the accuracies of decision tree is comparable to other classification techniques, especially for some simple data sets; it's quite robust to noise, especially when methods for avoiding overfitting [1]; redundant attributes does not adversely influence the accuracies of decision tree algorithm, but irrelevant attributes may impact greatly on effect of decision tree (related work has been processed in data pre-processing) [2]; splitting of decision tree should be disallowed when the number of records falls below a certain threshold to avoid generating data fragmentation.

The basic algorithm for decision tree is a greedy algorithm that constructs the tree in a top-down manner. The basic strategy is mainly listed as follows:

·The tree starts as a single node representing the training samples;

·If the samples are all of the same class, then the node becomes a leaf and is labeled with that class;

·Otherwise, the algorithm uses an entropy-based measure known as information gain to select the attribute that will best separate the samples into individual classed. Continuous-valued attributes must be discretized [5];

·A branch is created for each known value of the test attribute, and samples are partitioned;

·The algorithm uses the same process recursively to form a decision tree for the samples at each partition.

The decision tree model we use is mainly based on ID3 algorithm and partly C4.5 [3].

### 5.1.2 Sample selection and algorithm

The training samples consist of 40700 data with 42 dimensions including the ID dimension and the classification dimension. We divide the training sample into 2 classed labeled with class 1(Target_Flag=1) and class 2(Target_Flag=0). We use progressive sampling to get suitable amount of samples. The accuracy of prediction model increases when the amount of samples increasing. And we use the amount of samples when the accuracy of prediction model tends to be stable. Considering our task is to predict the sample of class 1, it's more unacceptable to mistake a sample in class 1 for a sample in class 2 than the reverse situation. Since the amount of sample in class 1 is much less than that of class 2 (700:40000), we increase the amount of samples in class 1 for each sampling. Oversampling is aimed at sampling for class 1 while undersampling is aimed at sampling for class 2.

The description of algorithm is in Table 2, Table 3 and Table 4:

**Table 2 The Generate_Decision_Tree algorithm**

**Algorithm:** Generate_Decision_tree()

**Input:** training samples presented by discretized value, the set of candidate attributes attribute_list

**Output:** a decision tree

**Method:**

1) Create a node N;

2) If samples are all of the same class C, label N with C;

3) If attribute_list is empty , label N as with class UnBalanceChoose;

4) Select the best partitioning attribute M according to (information gain/gain rate): DecisionDivChoose(WhereSen, attributName), then label N;

5 ) If (information gain/gain rate)<x,

Generate the leaf node labeled class UnBalanceChoose

Else

for each set of record *si* generated by distinct partition mi of attribute M of node N

Find the class of new node recursively;

If si>n //n is the threshold for pruning)

Generate_Decision_tree();

Else

Label n with class UnBalanceChoose

**Table.3 The UnBalance_Choose algorithm**

**Algorithm**: UnBalanceChoose()

1) Count the number of occurrence of each class and sort them

If p(+/t)>c(-,+)/(c(-,+)+c(+,-)) // c(-,+) is the cost of generating a False Positive, c(+,-) is the cost of generating a False Negative. p(i/t) stands for the percentage of node t in class i

2) Label t with class+

3) Else

4) Label t with class negative

5) Return the class of t

**Table.4 The Generate_decision_tree_predict algorithm**

**Algorithm**: Generate_decision_tree_predict(DataBase,TableName)

1) Read the model of decision tree

2) Read the set of prediction

3) For each row i in the set of prediction

4) From the root node

While node j is not the leaf

Inquire the value of record i attributvalue according to the name of attribute in node j;

Find the child node of j through attributvalue;

j=j.child

End

i.div=j.div

We test 5 sets of samples for the same size with the thought of Bagging, and results of experiments are as Table 5 (the number of first row stands for TN, second is FP, third is FN, and forth is TP):

**Table 5. Results of experiments for decision tree**

| UNnewdivatt3669a | UNnewdivatt3669b | UNnewdivatt3669c | UNnewdivatt3669d | UNnewdivatt3669e |
|---|---|---|---|---|
| 1094 | 1042 | 1102 | 1059 | 1092 |
| 875 | 927 | 867 | 910 | 877 |
| 17 | 15 | 10 | 12 | 11 |
| 14 | 16 | 21 | 19 | 20 |

(Sample of 3669 data)

| doublenewdivatt3669a | doublenewdivatt3669b | doublenewdivatt3669c | doublenewdivatt3669d | doublenewdivatt3669e |
|---|---|---|---|---|
| 1448 | 1457 | 1435 | 1432 | 1717 |
| 521 | 512 | 534 | 537 | 252 |
| 22 | 14 | 15 | 18 | 22 |
| 9 | 17 | 16 | 13 | 9 |

(Sample of 3669 data without Un-Balance-Choose() )

| UNdoublenewdivatt5669a | UNdoublenewdivatt5669b | UNdoublenewdivatt5669c | UNdoublenewdivatt5669d | UNdoublenewdivatt5669e |
|---|---|---|---|---|
| 1609 | 1602 | 1568 | 1583 | 1500 |
| 360 | 367 | 401 | 386 | 469 |
| 24 | 19 | 25 | 20 | 21 |
| 7 | 12 | 6 | 16 | 10 |

(Sample of 5669 data with decision tree pruning)

| UNdoublenewdivatt20 | UNdoublenewdivatt20 | UNdoublenewdivatt20 | UNdoublenewdivatt20 | UNdoublenewdivatt20 |
|---|---|---|---|---|

| 669a | 669b | 669c | 669d | 669e |
|---|---|---|---|---|
| 1751 | 1780 | 1743 | 1780 | 1765 |
| 218 | 189 | 226 | 189 | 204 |
| 26 | 26 | 27 | 26 | 25 |
| 5 | 5 | 4 | 5 | 6 |

(Sample of 20669 data)

It's concluded that a better prediction result could be found in the prediction set with 20669 samples and decision tree pruning.

### 5.2 Artificial neural network

5.2.1 Motivation

At present, Back-Propagation Network and its variation are prevailing in most ANN model for practical application, which is also the nuclear part of Forward Network [6].

The algorithm of BP could find more optimal solution when momentum is used; while the algorithm of BP could shorten the training time when adaptive learning is used. Trainbpx function trains the multilayer forward network with the approaches mentioned above.

Elman network consists of a tansig layer and a pure linear output layer. Tansig layer receives the feedback signal from itself and input it to network, and pure linear layer receives the input from tansig layer. Elman network could perform any one of the limited functions well, and identify or generate time mode and space mode by learning from its feedback [7].

Levenberg-Marquardt algorithm is faster than the gradient descending used by trainbp function and trainbpx function, but it needs more memory.

In practice application, the original BP algorithm fails to be the most suitable approach. Trainbpx algorithm (also known as Fast BP algorithm) improves the efficiency of learning and the reliability of algorithm by adopting momentum and adaptive adjustment. Therefore, we choose Levenberg-Marquardt algorithm and Fast BP algorithm [6,7].

#### 5.2.2 Design of ANN structure

**1) Number of network layers**

It has been proved that a network with deviation, a S hidden layer at least and a linear output layer could approximate to any rational function. Adding layers could further cut down the number of errors but complicate the structure of network and increase the training time. Error precision could be improved through adding the number of neuron in hidden layers. It's easier to observe and adjust training effect.

**2) Number of neuron in hidden layer**

Adding the number of neuron in hidden layer could improve the precision of training, and it's easier to be realized than adding more hidden layers. Comparing with the training result for different number of neuron, we set 5 neurons in the first hidden layer.

**3) Combination of transfer function in hidden layer**

We train 6345 random samples and analyze the result based on the prediction by the combination of different transfer function on hidden layer with 2000 data (there are 30 data for Target_Flag=1). Results have been shown in Fig. 2. The gray area indicates comparatively better results. So we decide to use the combination of Levenberg-Marquardt algorithm and L-P hidden transfer function to make the further prediction.

| Levenberg-Marquardt | T-P | T-L | L-L | L-P | P-P |
|---|---|---|---|---|---|
| ff | 1307 | 1390 | 1341 | 1446 | 1233 |
| fs | 662 | 579 | 628 | 523 | 736 |
| sf | 13 | 13 | 13 | 13 | 13 |
| ss | 18 | 18 | 18 | 18 | 18 |
| | | | | | |
| Fast Adaptive BP algorithm | T-P | T-L | L-L | L-P | P-P |
| ff | 1209 | 1283 | 1200 | 1381 | 1332 |
| fs | 760 | 686 | 769 | 588 | 637 |
| sf | 14 | 11 | 13 | 12 | 13 |
| ss | 17 | 20 | 18 | 19 | 18 |

**Fig. 2 Comparison of two ANN algorithms**

( ff-TN; fs-FP; sf-FN; ss-TP;  T-tansig; L-logsig; P-purelin )

**4) Selection of learning rate and initial weight**

Since the system is non-linear, the selection of initial value plays an important role in achieving part minimum, function convergence and training time.

The rate of learning decides weight variation in each repetitive training. Fast rate may result in instability of system. Although slow rate will extend training time, it could make sure that the error of network will be limited in certain range. A sensible solution is to adopt changeable adaptive learning rate in different learning phase (e.g. trainbpx).

We use 5 random samples (22367 for each sample) to predict 2000 data (|Target_Flag=1|=30) to decide the initial weight and learning rate. Prediction results are showed in Fig. 3.

| divatt40700learn1(sample.1) | | | | | |
|---|---|---|---|---|---|
| threshold | 0.2 | 0.18 | 0.15 | 0.13 | 0.1 |
| ff | 1569 | 1425 | 1268 | 1166 | 1025 |
| fs | 400 | 544 | 701 | 803 | 944 |
| sf | 18 | 18 | 14 | 10 | 7 |
| ss | 13 | 13 | 17 | 21 | 24 |

| divatt40700learn2(sample.2) | | | | | |
|---|---|---|---|---|---|
| threshold | 0.2 | 0.18 | 0.15 | 0.13 | 0.1 |
| ff | 1629 | 1455 | 1301 | 1045 | 931 |
| fs | 340 | 514 | 668 | 924 | 1038 |
| sf | 18 | 16 | 15 | 9 | 9 |
| ss | 13 | 15 | 16 | 22 | 22 |

| divatt40700learn3(sample.3) | | | | | |
|---|---|---|---|---|---|
| threshold | 0.2 | 0.18 | 0.15 | 0.13 | 0.1 |
| ff | 1498 | 1388 | 1261 | 1060 | 725 |
| fs | 471 | 581 | 708 | 909 | 1244 |
| sf | 18 | 16 | 12 | 7 | 5 |
| ss | 13 | 15 | 19 | 24 | 26 |

| divatt40700learn4(sample.4) | | | | | |
|---|---|---|---|---|---|
| threshold | 0.2 | 0.18 | 0.15 | 0.13 | 0.1 |
| ff | 1635 | 1577 | 1246 | 1079 | 1005 |
| fs | 334 | 392 | 723 | 890 | 964 |
| sf | 17 | 16 | 13 | 10 | 9 |
| ss | 14 | 15 | 18 | 21 | 22 |

| divatt40700learn5(sample.5) | | | | | |
|---|---|---|---|---|---|
| threshold | 0.2 | 0.18 | 0.15 | 0.13 | 0.1 |
| ff | 1541 | 1443 | 1306 | 1056 | 759 |
| fs | 428 | 526 | 663 | 913 | 1210 |
| sf | 16 | 14 | 14 | 9 | 6 |
| ss | 15 | 17 | 17 | 22 | 25 |

**Fig. 3 Prediction results**

## 6 Analysis and Evaluation

We use the ROC (Receiver Operating Characteristic curve) [4] to evaluate the accuracy of each prediction model in Fig.4. The closer the area is to 0.5, the lousier the model, and the closer it is to 1.0, the better the model. We find although the result of ANN is better than that of Decision Tree according to the area under the ROC curve, there are still a lot of values predicted incorrectly. To reduce the rate of errors as low as possible, we consider the prediction result of decision tree partly and combine these two kinds of prediction result to make a more precise prediction. We compare the previous result with the combination result, and find the result of combination model is better

than that of separate sample when the threshold ranges from 0.05 to 0.1.

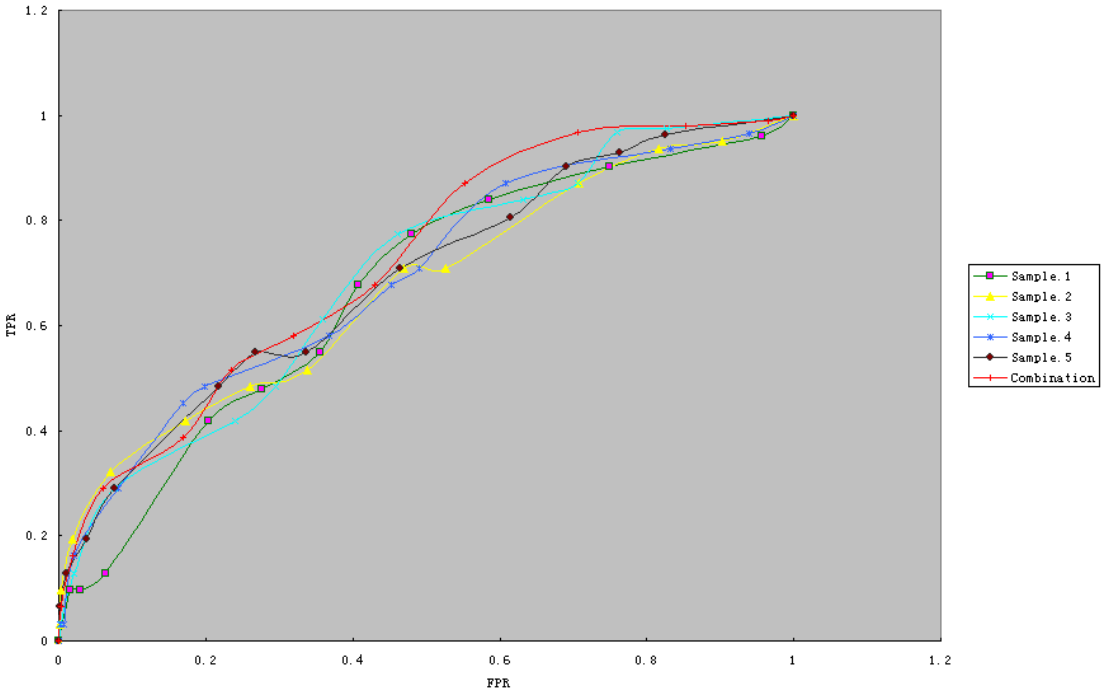


**Fig.4 The ROC curve**

## 7 Conclusions

The cross-selling business problem is an interesting challenge for data mining. There have been many effective techniques to analyze and predict this kind of dataset. But we can not confirm that a certain technique is the best solution to this problem. We can get some useful information and prediction model from a variety of available approaches. In this paper, we choose decision tree and artificial neural network as mining tools. For each technique, different algorithm and sample set could be used to make more precise prediction. So we use the combination of Levenberg-Marquardt algorithm and L-P hidden transfer function with different threshold value in ANN and a hybrid sampling method in decision tree. It's effective for further prediction to compare the advantage and disadvantage of each technique and combine their prediction results properly.

There are some useful business insights that could be interpreted from our scoring model results: 1) we should pay more attention to these attributes whose information gain is greater than a certain threshold, since these attributes can make much more contribution to effective prediction; 2) we can properly remove some customers who are considered as customers not to take up a home loan in a certain branch of the decision tree, since True Negative Fraction always takes the mode in prediction.

**References**

1. J.R. Quinlan. Induction of decision trees. Machine Learning, 1:81-106, 1986.
2. J.R Quinlan. Learning efficient classification procedures. In Machine Learning: An Artificial Intelligence Approach, 1983.

3. RRastogi and K.Shim. PUBLIC: A Decision Tree Classifier that Integrates Pruning and Building. In Proc. of VLDB, 1998.
4. Bradley, A. P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms 30(7) (1997) 1145-1159.
5. J. Dougherty, R. Kahove, and M. Sahami. Supervised and unsupervised discretization of continous features. In Proc. of Machine Learning, 1995.
6. Yi S, et al. Global Optimization for NN Training. IEEE Computer, 1996, 3:45-54
7. Pingfan Yan, Changshui Zhang. ANN and Simulation Evolutionary Computation. The Tsinghua Press, 2000