The Application of AdaBoost in Customer Churn Prediction

Shao Jinbo[1], Li Xiu[2], Liu Wenhuang[3]

Automation Department, Tsinghua University, Beijing, 100084, China
sjb@mails.tsinghua.edu.cn

ABSTRACT

Since attracting new customers is known to be more expensive, the enhancement of existing relationships is of pivotal importance to companies. Therefore, as part of the customer relationship management (CRM) strategy, predicting customer churn and improving customer retention have attracted more and more attention. Being aware of the defection prone customers beforehand, companies could react in time to prevent the churn by offering the right set of products, modifying the sales strategy and providing customized services. Therefore, high predictive performance could ultimately lead to profit increasing for companies.

In this paper, we use the AdaBoost which is a main branch of boosting algorithms to predict the customer churn. We have implemented three different boosting schemes: Real AdaBoost, Gentle AdaBoost and Modest AdaBoost. Applied to a credit debt customer database of an anonymous commercial bank in China, they are proven to significantly improve prediction accuracy comparing with other algorithms, like SVM. The assessment and comparison of these algorithms are made to analyze the traits of them. Data processing and sampling scheme are also detailed in this paper.

Keywords: Customer Churn, Prediction, boosting, AdaBoost

1. INTRODUCTION

This paper studies the customer churn that is a hot topic in CRM and also the most important issues in enterprises. Customer churn – the propensity of customers to cease doing business with a company in a given time period – has become an important problem for many firms which include publishing, investment services, insurance, electric utilities, health care providers, credit card providers, banking, Internet service providers, telephone service providers, online services, and cable services operators[1]. Obviously, customer churn figures directly in how long a customer stays with a company, and in turn the customer's lifetime value to that company. By analyzing the current of a customer's lifetime profit to a company[2], it is easy to find that most of the company's profits are contributed by frequent customers and attracting new customers is more expensive than retaining the existing ones. Therefore, the enhancement of relationships with existing customers is of pivotal importance to companies. Being aware of the defection prone customers beforehand, companies could react in time to prevent the churn. So, customer churn prediction is the first and also a very important step to prevent customer churn. What we try to do is to identify in advance those customers who are likely to churn at some later date. The company then can target these customers with special programs or incentives to forestall the customer from churning.

The most widely used model for predicting the customer churn is the binary classification model. The customers can be classified into two categories: going to churn or not. Many methods and algorithms are used to solve this problem, such as classification tree[3], neural network[4] and genetic algorithms[5]. Decision tree based algorithms can uncover the classification rules for classifying records with unknown class membership. Nevertheless, when decision tree based algorithms are extended to determine the probabilities associated with such classifications[6], it is possible that some leaves in a decision tree have similar class probabilities. Neural networks can determine a probability for a prediction with its likelihood. However, comparing with decision tree based algorithms these algorithms do not explicitly express the uncovered patterns in a symbolic, easily understandable way. Genetic algorithms can produce accurate predictive models, but they cannot determine the likelihood associated with their predictions. This prevents these techniques from being applicable to the task of predicting churn, which requires the ranking of customers according to their likelihood to churn[7].

Except algorithms above, some scholars put forward some other methods to predict the churn. Luo[8] applied Bayesian multi-net classifier in customer modeling of telecommunications CRM and got effective results. Zhao[9] introduced an improved one-class SVM and tested it on a wireless industry customer churn data set. Ding[10] studied the application of sequential pattern association analysis in the prediction of customer churn in banking. Lu[11] used survival analysis to model customer lifetime value which is a powerful and straightforward measure that synthesizes customer profitability and churn (attrition) risk at individual customer level. Some other scholars also use some combination methods to predict the churn[7][12]. All of these have made good attempts in predicting the churn and ultimately increasing the customers' value for the companies.

Lemmens and Croux[13] are the first who applied the ensemble learning algorithm in prediction of customer churn. They tested bagging and stochastic gradient boosting[14], one of the most recent boosting variants, on

1-4244-0885-7/07/$20.00 © 2007 IEEE.

a customer database of an anonymous U.S. wireless telecom company and reported a significant predication accuracy improvement. Our work is to put the research one step forward. We focus on the boosting and apply three different boosting schemes to a credit debt customer database of an anonymous commercial bank in China. Data processing and sampling scheme are detailed in the section after next. The assessment and comparison of these algorithms are made to analyze the traits of them. Ultimately, we draw a conclusion.

2. METHODOLOGY

Boosting is one of the most important recent developments in classification methodology. It is a technique of combining a set of weak classifiers to form one high-performance prediction rule (a powerful "strong" classifier or "committee"). It works by sequentially applying a classification algorithm to re-weighted versions of the training data and then taking a weighted majority vote of the sequence of classifiers thus produced.

The first practical boosting algorithm, called AdaBoost, was proposed by Freund and Schapire[15] in 1996. AdaBoost is adaptive in that it adapts to the error rates of the individual weak hypotheses. This is the basis of its name — "Ada" is short for "adaptive."[16]

AdaBoost has many advantages. It is fast, simple and easy to program. It has no parameters to tune (except for the number of round T). It requires no prior knowledge about the weak learner and so can be flexibly combined with any method for finding weak hypotheses. Finally, it comes with a set of theoretical guarantees given sufficient data and a weak learner that can reliably provide only moderately accurate weak hypotheses. This is a shift in mind set for the learning-system designer: instead of trying to design a learning algorithm that is accurate over the entire space, we can focus on finding weak learning algorithms that only need to be better than random [16].

In 1999, Schapire and Singer[17] studied boosting in an extended framework in which each weak hypothesis generates not only predicted classifications, but also self-rated confidence scores which estimate the reliability of each of its predictions. They also discussed some essential questions in boosting. Then they gave an improved generalized version of AdaBoost. The algorithm takes as input a training set $(x_1, y_1), ..., (x_m, y_m)$ where each x_i belongs to some domain or instance space X, and each label y_i is in the label set $Y = \{-1, +1\}$. AdaBoost calls a given weak or base learning algorithm repeatedly in a series of rounds $t = 1, ..., T$. One of the main ideas of the algorithm is to maintain a distribution or set of weights over the training set. The weight of this distribution on training example i on round t is denoted $D_t(i)$. Initially, all weights could be set equally, but on each round, the weights of incorrectly classified examples are increased so that the weak learner is forced to focus on the hard examples in the training set. The weak learner's job is to find a weak hypothesis $h_t : X \to \mathbb{R}$ appropriate for the distribution D_t. The goodness of a weak hypothesis is measured by its error:

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i: h_t(x_i) \neq y_i} D_t(i) \tag{1}$$

So the steps of the generalized AdaBoost algorithm are:

For $t = 1, ..., T$:

-Train weak learner using distribution D_t.

-Get weak hypothesis $h_t : X \to \mathbb{R}$ with error

$$\varepsilon_t = \sum_{i: h_t(x_i) \neq y_i} D_t(i) \tag{2}$$

-Choose

$$\alpha_t \in \mathbb{R} \tag{3}$$

-Update:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \tag{4}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

-Output the final hypothesis:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{5}$$

And then they proved that, in order to minimize training error, a reasonable approach might be to greedily minimize the bound given in the theorem by minimizing Z_t on each round of boosting. It can be verified that Z is minimized when

$$\alpha = \frac{1}{2} \ln\left(\frac{W_{+1}}{W_{-1}}\right) \tag{6}$$

$$W_b = \sum_{i: y_i h_t(x_i) = b} D(i) \tag{7}$$

So they replaced the α_t in the generalized AdaBoost steps with the new $\alpha_t = \frac{1}{2} \ln\left(\frac{W_{+1}}{W_{-1}}\right)$ to form a new AdaBoost algorithm — the Real AdaBoost. The Real AdaBoost algorithm uses class probability estimates W_b

to construct real-valued contributions $\alpha_t h_t(x)$. And it is usually treated as a basic "hardcore" boosting algorithm.

In 2000, Friedman, Hastie and Tibshirani[18] put forward another improved AdaBoost algorithm, called Gentle AdaBoost. Here the update is

$$\alpha_t = \frac{1}{2}\ln(\frac{W_{+1} - W_{-1}}{W_{+1} + W_{-1}}) \tag{8}$$

rather than $\alpha_t = \frac{1}{2}\ln(\frac{W_{+1}}{W_{-1}})$. This makes the Gentle AdaBoost have better generalizing ability so as to solve the overfitting problem and noise sensitive problem that former AdaBoost algorithms are facing. Some empirical evidence suggests that this more conservative algorithm has similar performance to the Real AdaBoost, and often outperforms it, especially when stability is an issue.

In 2005, A. Vezhnevets and V. Vezhnevets[19] introduced Modest AdaBoost algorithm. They used

$$\overline{D}_t(i) = (1 - D_t(i))\overline{Z}_t \tag{9}$$

to construct the new

$$\alpha_t = W_{+1}(1 - \overline{W_{+1}}) - W_{-1}(1 - \overline{W_{-1}}) \tag{10}$$

$$\overline{W_b} = \sum_{i: y_i h_t(x_i) = b} \overline{D}(i) \tag{11}$$

They applied the new algorithm to UCI Machine Learning Repository database and compared the result with using Gentle AdaBoost. In some datasets of the UCI database, the Modest AdaBoost outperforms the Gentle AdaBoost in error rate and seems to be more stable — is resistant to overfitting more. The drawback of Modest AdaBoost is that training error decreases much slower then in Gentle AdaBoost scheme and often does not reach zero point. Because they decreased weak classifiers' contribution if it works "too good" on data that had been already correctly classified with high margin. This makes the algorithm better generalizing ability but lower learning speed.

It is known that there is no algorithm fit for all datasets. So, in our experiment, we will apply these three representative AdaBoost algorithms to our dataset to see which one is the most suitable algorithm for our problem. And we will use stumps as weak classifier for both methods. This choice was made because stumps are considered to be the "weakest of all" among commonly used weak learners, so we hope that using stumps lets us investigate the difference in performance resulting from different boosting schemes.

3. DATA PROCESSING AND SAMPLING SCHEME

Our study is performed on a database provided by an anonymous bank in China. The database has nearly 20,000 observations in total. We select 1,524 observations from the database to form our experiment dataset. The observations that are lack of important attributes or lack of too many attributes (more than 30% of the total) are excluded.

We select the attributes (variables) of the customers (observations) after fixing the observations. The variable selection is done by first excluding the attributes used for management of the bank, such as Customer ID. Then we exclude all variables containing more than 30% of missing values. We retain 19 variables, including customer demographics (e.g. the number of children in the household, or the education level of the customer) variables, behavioral (e.g. the type of the customer's debt, the type of hypothecation, or the term of the debt), and company interaction (e.g. the number of exceeding time limit times).

We also need to translate the character attributes into numbers. For the attribute whose available values have trend (e.g. the education level of the customer), we can translate the values of the attribute into numbers with trend (e.g. education level low equals 1, middle equals 2 and high equals 3). For the attribute whose available values have no trend (e.g. the type of the customer's debt), we should extend this attribute (variable) to several variables (shown in Figure 1).

Customer	Debt Type
Customer A	Type 1
Customer B	Type 2
Customer C	Type 4

→

Customer	Type 1	Type 2	Type 3	Type 4
Customer A	1	0	0	0
Customer B	0	1	0	0
Customer C	0	0	0	1

Figure 1

The handling of missing values is operated differently for the continuous and the categorical predictors. For the continuous variables, the missing values are imputed by the mean of the non-missing ones. For categorical predictors, an extra level is created for each of them, indicating whether the value was missing or not.

At last we define the meaning of churn. The staffs of the bank have already classified and rated these customers according to customers' credit by their experiences in banking. We define the customers whose credit rates are "low" as churners. The churners are around 5% of the total customers. The churn response (customer label) is coded as a variable with y = 1 if the customer churns, and y = −1 otherwise.

Now we have the full experiment dataset that has 1524 customers, 27 predictor variables and the label variable. Then we divide the full experiment dataset into two different datasets averagely. The first one containing half of the total observations is used for training the

classifiers and the other observations are used for testing and estimating the classifiers.

As we could see, customers' defection is still — statistically speaking — a rare event (around 5% of the total customers, even less in some other industries, e.g. 1.5% in wireless industry). Consequently, when the churn predictive model is estimated on a random sample of the customers' population, the vast majority of non-churners in this proportional training dataset (i.e. the number of churners in this randomly drawn sample is proportional to the real-life churn proportion) will dominate the statistical analysis, which may hinder the detection of churn drivers, and eventually decrease the predictive accuracy. So what we do is re-sampling the churners in the training set to make a relatively balanced training set. This will increase the size of the training set and also the computational work amount. But we could achieve it by setting the initial distribution of the training examples, if the weak learner is an algorithm that can use the weights distribution on the training examples. In this way, we can keep the computational work amount the same with of proportional training set[15]. Both sampling schemes are assessed in next section. And we will give an empirical conclusion.

4. RESULTS AND DISCUSSION

Our experiments are done in Matlab 2006b Edition with the help of GML AdaBoost Matlab Toolbox. First, let's see the error rates of these three AdaBoost algorithms in two sampling schemes. The error rate here means the percentage of incorrectly classified observations in the validation dataset.

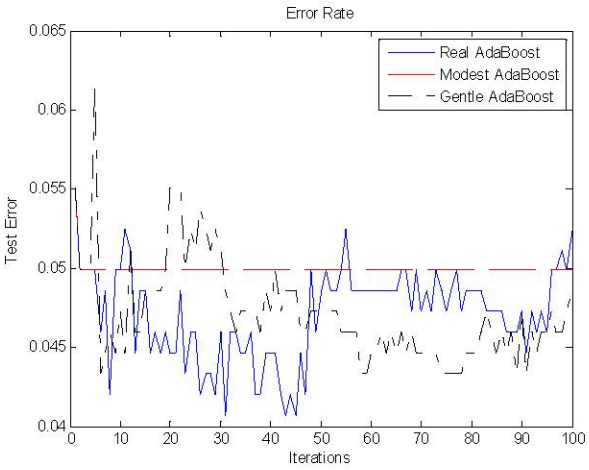

Figure 2 Error rate with the proportional training set

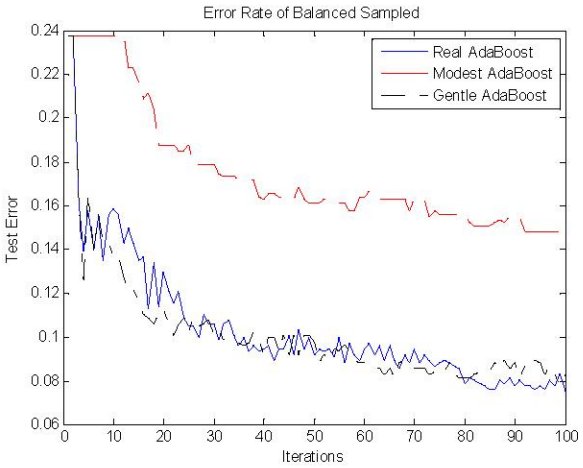

Figure 3 Error rate of balanced sampled

Fig.2 and Fig.3 show the error rates of the three AdaBoost algorithms in two different sampling schemes. It can be seen that the lowest error rate is about 4.5%. This seems to be an excellent performance. But if we take the badly unbalanced dataset (about 5% are churners) into account, we will find that such a rule maybe don't isolate any potentially riskiest customers. So, for rare events, the error rate is often inappropriate.

So, we should choose another assessment criterion. The *lift* is a usually used criterion in prediction. And what we really care about is who the riskiest customers are. So, we use the top-decile lift as the assessment criterion. The top-decile lift focuses on the most critical group of customers regarding their churn risk. The top-decile lift equals the proportion of churners in this risky segment, $\hat{\pi}_{10\%}$, divided by the proportion of churners in the whole validation set, $\hat{\pi}$:

$$Top-decile\ lift = \frac{\hat{\pi}_{10\%}}{\hat{\pi}} \tag{12}$$

The higher the top-decile lift, the better the classifier. And the score values of the churn risk can be obtained in these algorithms by making the final hypothesis:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{13}$$

The top 10% riskiest customers is also potentially an ideal segment for targeting a retention marketing campaign.

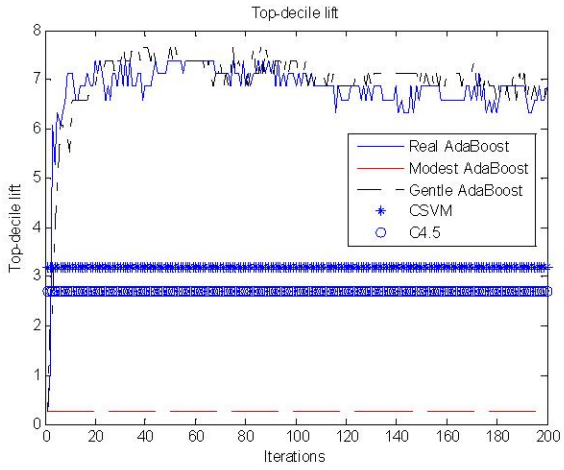

Figure 4 Top-decile lift with the proportional training set

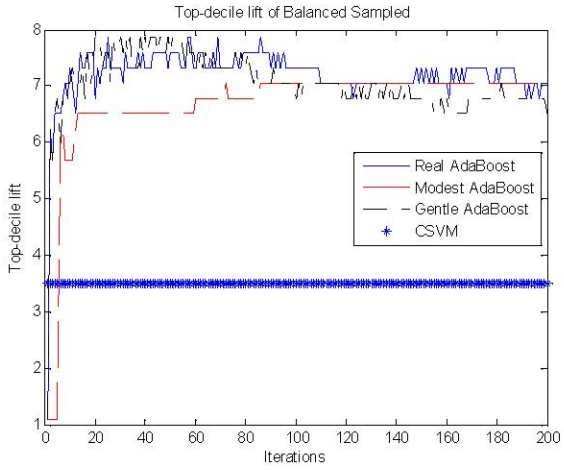

Figure 5 Top-decile lift of balanced sampled

Fig.4 and Fig.5 show the top-decile lift of the three AdaBoost algorithms in two different sampling schemes. Comparing with C-SVM (C-SVM in balanced sampled is CWC-SVM[9]) and C4.5, AdaBoost have a relatively better performance and really improve a lot — twice or more. That means there are nearly 80% of the total potential churners in our predicted 10% riskiest customers.

Now let's compare these three AdaBoost algorithms with each other. We can see that Real AdaBoost and Gentle AdaBoost have close performances on our dataset with both sampling schemes and in both assessment criteria. The one with higher learning speed, less error rate or higher lift value is usually more overfitting. Just in Fig.4, Gentle AdaBoost performs very slightly better than Real AdaBoost. Modest AdaBoost seems to have some trouble in our dataset. It is too "modest" to fit our heavily unbalanced dataset. It can't boost at all unless we resample the data to form a balanced training set. With balanced training set, Modest AdaBoost is still learning slower than the other two algorithms. But it resists the overfitting problem very well, which can be seen in Fig.5.

The two different sampling schemes give us different results as well as some clue. Balanced sampling scheme make a big drawback in error rate for all three algorithms. But it improves the top-decile lift of Real and Gentle AdaBoost slightly and top-decile lift of Modest AdaBoost greatly. So, if we take top-decile lift as the prior assessment criterion, balanced sampling scheme is a good choice to solve the heavily unbalanced problem of training set. Actually, balanced sampling scheme "helps" Modest AdaBoost a lot for our dataset, even in error rate. It puts Modest AdaBoost to start to boost, shown in Fig.3.

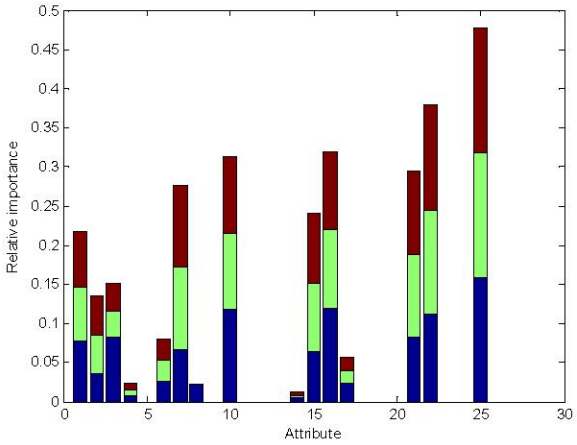

Figure 6 Weights of each attribute in three algorithms

There is also another advantage of AdaBoost. It can indicate the potential rules of the classification process. Each weak learner uses an attribute (variable) to classify the dataset. The attribute number can be obtained easily. And the absolute values of the weights of these weak learners represent the "confidence" of the weak learners, i.e. the attributes. So, we can find which attribute is the most powerful influencing factor to the classification. Fig.6 shows the weights of each attribute in the three AdaBoost algorithms. It could be seen that the top three influencing factors are the amount of the debt, the customer's duty level and the type of repayment. And we can also find that different AdaBoost algorithms have nearly the same result in choosing attributes.

5. CONCLUSION

To sum up, AdaBoost algorithms make a really good performance in predicting the customer churn. They can not only determine a probability for a prediction with its likelihood, but also explicitly indicate the potential rules of the classification process.

Real and Gentle AdaBoost are adaptable to the heavily unbalanced churner dataset even with the "weakest" learner — stump. Modest AdaBoost are too conservative to adapt to the dataset. But balanced sampling scheme can improve the performance of each algorithm in top-decile lift with drawback in error rate. Modest

AdaBoost shows its resistance ability to overfitting with balanced sampling scheme.

There is still a lot of further work to do. New improved AdaBoost algorithms are put forward ceaselessly. We could keep searching for more suitable AdaBoost algorithms for customer churn prediction. And we could also try some other weak learners in further experiments. Whatever, we hope our work helpful for companies to better identify the riskiest customers' segment in terms of churn risk, and ameliorate the retention strategy. Ultimately, they could reduce the losses caused by the churn.

ACKNOWLEDGEMENT

The authors are very grateful to the anonymous bank that supplied the data to perform the analysis and to the managers who help us a lot by sharing their insights and expertise.

REFERENCES

[1] S Neslin, S Gupta, W Kamakura, J Lu, C Mason, "Defection Detection: Improving Predictive Accuracy of Customer Churn Models", *Working Paper, Teradata Center at Duke University*, 2004.3

[2] Wells, Melanie, "Brand ads should target existing customers" *Advertising Age*, pp26-47, 1993.

[3] CP Wei, IT Chiu, "Turning telecommunications call details to churn prediction: a data mining approach", *Expert Systems With Applications*, Vol. 23, Issue 2, pp103-112, 2002.

[4] Mozer, M. C., Wolniewicz, R., Grimes, D. B., etc, "Churn reduction in the wireless industry", *Advances in Neural Information Processing Systems*, pp935-941, 2000(12).

[5] Eiben, A.E.; Koudijs, A.E.; Slisser, F. Source, "Genetic modeling of customer retention", *Lecture Notes in Computer Science*, Vol. 1391, pp178, 1998

[6] J. R. Quinlan, "Decision trees as probabilistic classifiers", *Proc. 4th Int. Workshop Machine Learning*, pp31–37, Irvine, CA, 1987.

[7] Wai-Ho Au, Keith C. C. Chan, and Xin Yao, "A Novel Evolutionary Data Mining Algorithm with Applications to Churn Prediction", *IEEE Transactions on Evolutionary Computation*, Vol. 7, Issue 6, pp532-545, 2003

[8] LUO Ning, MU Zhichun, "Bayesian Network Classifier and Its Application in CRM", *Computer Application*, Vol. 24, No. 3, pp79-81, 2004

[9] Y Zhao, B Li, X Li, W Liu, S Ren, "Customer Churn Prediction Using Improved One-Class Support Vector Machine", *Lecture Notes in Computer Science*, Vol. 3584, pp300-307, 2005

[10]Ding-An Chianga, Yi-Fan Wangb, Shao-Lun Leea, Cheng-Jung Lina, "Goal-oriented sequential pattern for network banking churn analysis", *Expert Systems with Applications*, Vol. 25, Issue 3, pp293–302, 2003.

[11]Lu Junxiang, "Modeling Customer Lifetime Value using Survival Analysis – an Application in the telecommunications Industry", *SAS Institute Paper* 120 – 28, 2001

[12]Louis Anthony Cox, Jr. "Data Mining and Causal Modeling of Customer Behaviors", *Telecommunication Systems*, Vol. 21, pp349–381, 2002

[13]A Lemmens, C Croux, "Bagging and Boosting Classification Trees to predict churn", *Journal of Marketing Research*, Vol. 43, No. 2, pp276-286. 2006

[14]Friedman, Jerome H. "Stochastic Gradient Boosting", *Computational Statistics and Data Analysis*, Vol. 38, Issue 4, pp367-378. 2002

[15]Y Freund and R. E. Schapire, "Game theory, on-line prediction and boosting", *In Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pages 325–332, Desenzano del Garda, Italy, 1996.

[16]Yoav Freund, Robert E. Schapire, "A Short Introduction to Boosting" *Journal of Japanese Society for Artificial Intelligence*, Vol. 14, No. 5, pp771-780, September, 1999.

[17]R.E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions", *Machine Learning*, Vol. 37, No. 3, pp297-336, December 1999.

[18]Jerome Friedman, Trevor Hastie, and Robert Tibshirani, "Additive logistic regression: A statistical view of boosting", *The Annals of Statistics*, Vol. 28, No. 2, pp337–374, 2000.

[19]A. Vezhnevets and V. Vezhnevets, "Modest AdaBoost – teaching AdaBoost to generalize better" *Graphicon* 2005.