Defection Detection Analysis Based on Time-Dependent Data

Hee Seok Song[a], Jae Kyeong Kim[b,*], Kyung Hee Chae [c]

[a] Graduate School of Management, Korea Advanced Institute of Science and Technology
207-43 Cheongryangri-Dong, Dongdaemun, Seoul, 130-012, South Korea
Tel : 82-2-958-3670 Fax : 82-2-958-3604 C.P : 019-9146-9864
e-mail : hssong@kgsm.kaist.ac.kr

[b,c] School of Business Administration, KyungHee University
#1, Hoeki-Dong, Dongdaemoon, Seoul, 130-701, South Korea
Tel : 82-2-961-9355 Fax : 82-2-967-0788 C.P : 011-9940-7988
e-mail : jaek@khu.ac.kr
*Corresponding author

Abstract

Past and current customer behavior is the best predicator of future customer behavior. This paper introduces a procedure on personalized defection detection and prevention for an online game site. The basic idea for our defection detection and prevention is adopted from the observation that potential defectors have a tendency to take a couple of months or weeks to gradually change their behavior (i.e. trim-out their usage volume) before their eventual withdrawal. For this purpose, we suggest a SOM (Self-Organizing Map) based procedure to determine the possible states of customer behavior from past behavior data. Based on this representation of the state of behavior, potential defectors are detected by comparing their monitored trajectories of behavior states with frequent and confident trajectories of past defectors. The key feature of this study includes a defection prevention procedure which recommends the desirable behavior state for the next period so as to lower the likelihood of defection. The defection prevention procedure can be used to design a marketing campaign on an individual basis because it provides desirable behavior patterns for the next period. The experiments demonstrate that our approach is effective for defection prevention and efficient for defection detection because it predicts potential defectors without deterioration of prediction accuracy compared to that of the MLP (Multi-Layer Perceptron) neural network.

Keyword : Customer Relationship Management, Data Mining, Customer Defection, Self-Organizing Map, Association Rule Mining

1. Introduction

Customer retention is an increasingly pressing issue in today's competitive commercial arena (Ng and Liu 2000). A major technique for managing customer retention is data mining. Data mining has emerged over recent years as an extremely powerful approach to extract meaningful information from large databases and data warehouses. Many studies (Berson et al. 2000; Datta et al. 2000; Eiben et al. 1998; Mozer et al. 2000; Ng and Liu 2000; Raghavan et al. 2000; Smith et al. 2000; Yeo et al. 2001) have been conducted for customer retention to demonstrate the potential of data mining through experiments and case studies. Especially in the service industry such as telecommunications and internet service providing companies, there exist enormous customer behavior data because of its necessity for billing. This behavior data provides a good opportunity to predict future customer behavior (Datta et al. 2000; Mozer et al. 2000; Ng and Liu 2000; Raghavan et al. 2000; Song et al. 2001). The basic idea originates from the observation that potential defectors have a tendency to take a couple of months or weeks to gradually change their behavior (i.e. trim-out their usage volumes) before their eventual withdrawal. This gradual pulling out process offers the company the opportunity to detect the signals of defection. With this approach, we have several benefits compared with traditional defection detection studies based on the data mining techniques. First, we can take ample lead-time for defection prevention by providing early warnings before defections actually take place. Second, we can build a personalized defection detection system because customer behavior is very different for each individual in usage patterns. Finally, the key benefit is that our approach can provide not only a procedure for defection detection but also for defection prevention which recommends the desirable behavior state for the next period so as to lower the likelihood of defection.

However, defection detection based on customers' past multi-period behavior pattern is not easy because of the following reasons. First, multi-features about customer

behavior have to be used to predict the likelihood of defection. Second, these features may also be related to customer defection in a highly nonlinear way. To overcome these difficulties, we define the possible behavior state for a specific domain using SOM (Self-Organizing Map) and use this state representation in monitoring the trajectory of customer behavior. We adapted the SOM because it facilitates the understanding of complex behavior dynamics so that several variables and their interactions can be inspected simultaneously (Alhoniemi et al. 1999; Simula et al. 1999). Based on these state representations, the following procedures are developed: (1) mining the frequent and confident trajectories of customer behavior states over time for defectors and non-defectors, (2) monitoring user behavior and detecting the potential defectors by matching their trajectory of behavior states with discovered trajectories in (1), and (3) setting up an automated defection prevention procedure which assists building a personalized campaign plan by recommending the desirable behavior state for the next period that lowers the likelihood of defection.

2. Existing works in defection detection using data mining techniques

Many studies (Berson et al. 2000; Datta et al. 2000; Eiben et al. 1998; Mozer et al. 2000; Ng and Liu 2000; Raghavan et al. 2000; Smith et al. 2000; Yeo et al. 2001) have been conducted to detect potential defectors using data mining techniques. Most of their prediction models for defection are developed using demographic and customer profile information as independent features. In particular in service industries that can capture various customer behaviors for the purpose of billing such as telecommunications and internet service providing companies, customer behavior data is mainly used to gauge and predict the likelihood of defection (Datta et al. 2000; Mozer et al. 2000; Ng and Liu 2000; Raghavan et al. 2000). However, these studies only focus on providing defection detection. Although potential defectors have been identified, each customer will eventually defect without the guidance to control his/her undesirable behavior which leads to defection. But, we have not yet found existing works in the field of customer retention, which have handled a defection prevention procedure.

3. Background

In this section, we briefly describe the background of two data mining techniques called SOM and association rule mining which are prerequisite for our procedure.

3.1 SOM

SOM was developed in its present form by Kohonen (Kohonen 1990; Kohonen 1995; Kohonen et al. 1996; Simula et al. 1999) and thus they are also known as Kohonen Maps. SOM is able to map structured, high-dimensional data onto a much lower-dimensional array of neurons in an orderly fashion. This mapping tends to preserve the topological relationships of the input data. Topological preserving means that the data points lying near each other in the input space will be mapped onto nearby map units. Due to this topology preserving property, SOM is able to cluster input information and their relationships on the map. The SOM model is made up of two neural layers. The input layer has as many neurons as it has variables. The output layer contains neurons that are arranged in a rectangular pattern or hexagonal pattern, which is called "the map". Each neuron in the input layer is connected to each neuron in the output layer. Thus, each neuron in the output layer has the same number of connections to the number of input neurons. Each one of these connections has a synaptic weight associated with it. In SOM, each neuron is represented by a n-dimensional weight vector, $\mathbf{m} = [m_1, m_2, ..., m_n]$, where n is equal to the dimension of the input vectors.

SOM is trained iteratively. In each training step, one sample vector $\mathbf{x}$ from the input data set is chosen randomly and the distance between it and all the weight vectors of SOM is calculated using some distance measure, e.g. Euclidian distance. The neuron whose weight vector is closest to the input vector $\mathbf{x}$ is called the Best-Matching Unit (BMU) : $\|\mathbf{x} - \mathbf{m}_c\| = \min_i \{\|\mathbf{x} - \mathbf{m}_i\|\}$, where $\|..\|$ is the distance measure, and $\mathbf{m}_c$ is the weight vector of BMU. After finding the BMU, the weight vectors of the SOM are updated so that the BMU is moved closer to the input vector in the input space. The topological neighbors of the BMU are also treated in a similar way. This adaptation procedure stretches the BMU and its topological neighbors towards the sample vector. The update rule for the weight vector of unit i is $\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{ci}(t)[\mathbf{x}(t) - \mathbf{m}_i(t)]$, where t denotes time, $\mathbf{x}(t)$ is the input vector randomly drawn from the input data set at time t, and $h_{ci}(t)$ is the neighborhood kernel around the winner unit c at time t. The neighborhood kernel is a non-increasing function of time and of the distance of unit i from the winner unit c. It defines the region of influence that the input sample has on SOM. The procedure is repeated until complete training stops. Once the training is completed, the weights are fixed and the network is ready to be used. From then on, when a new pattern is presented, each neuron computes in parallel the distance between the input vector and the weight vector that it stores, and a competition starts that is won by the neuron whose weights are more similar to the input vector. SOM has proven to be a valuable tool in data mining, full-text mining and financial data analysis. It has also been successfully applied in various engineering applications in pattern recognition, image analysis, process monitoring and fault diagnosis (Vesanto 1999).

3.2 Association rule mining

A typical association rule has an implication of the form $A \Rightarrow B$ where A is an itemset and B is an itemset that contains only a single atomic condition. The *support*

of an association rule is the percentage of records containing itemsets A and B together. The *confidence* of a rule is the percentage of records containing itemset A that also contain itemset B. Support represents the usefulness of a discovered rule, and confidence represents the certainty of the detected association rule. Association rule mining finds all collections of items in a database whose confidence and support meet or exceed the prespecified threshold value. Apriori algorithm is one of the prevalent techniques used to find association rules (Agrawal et al. 1993; Agrawal and Srikant 1994). Apriori operates in two phases. In the first phase, all large itemsets are generated. This phase utilizes the downward closure property of support. In other words, if k size (or length k) itemset is a large itemset, then all the itemsets below (k-1) size must also be large itemsets. Using this property, candidate itemsets of length k are generated from the set of large itemsets of length (k-1) by imposing the constraint that all subsets of length (k-1) of any candidate itemset must be present in the set of large itemsets of length (k-1). The second phase of the algorithm generates rules from the set of all large itemsets. Please refer to the study of Agrawal et al. (1993) for more detail. In our suggested procedure, association rule mining is used to discover frequent and confident trajectories of customer behavior states for past defectors and non-defectors.

4. Domain

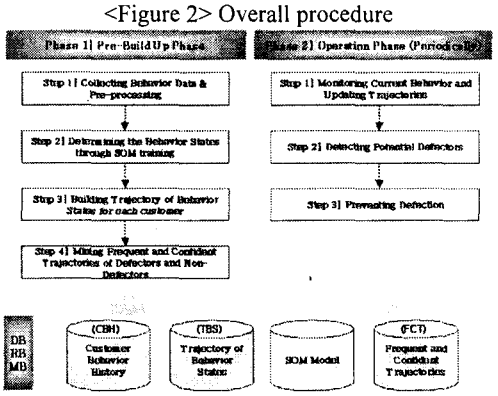
Before describing the detailed procedure, the domain and dataset for the case study are briefly explained in this section. The dataset is prepared from a Korean online game company. We sampled 255 customers and collected totally 114,736 transactions for those customers from various sources such as web log data and the transaction DB. We balanced percentage of actual defectors and non-defectors as 35 % and 65 %. The collected input data for defection prediction contains the totaled session length for a week, the total number of sessions for a week, the total number of access errors caused by congestion, and the average length per session. Customer profiles were maintained for the four input features during consecutive four weeks for each customer with an actual defection indicator. We selected a week as an appropriate time interval for aggregating the behavior data for our online game site because it seemed to be a reasonable trade-off between accuracy of prediction and lead-time for reaction. Also, we defined a defector as the customer who has no session time for one month, because there is no voluntary sign for termination of service in the internet game site. Final customer profiles are divided into a training set and a test set.

5. Proposed procedure

5.1 Overall procedure

The key concept of our proposed procedure for defection detection is adapted from the work on monitoring the state of an industrial process (Tryba and Goser 1991; Kasslin et al. 1992). The goal in process state monitoring is to develop a representation scheme of the state of an industrial process from process data and to use this representation in monitoring the current state of the process. For this purpose the SOM model is adapted because it facilitates understanding of a complex process, so that several variables and their interactions can be inspected simultaneously even when there exist non-linear dependencies between process variables (Alhoniemi et al. 1999; Simula et al. 1999). In this section, we present the overall procedure of our personalized procedure for defection detection and prevention with Figure 2.

<Figure 2> Overall procedure



As shown in Figure 2, the proposed methodology consists of two phases called the pre-build up phase and the operation phase. The pre-build up phase is performed once to build a reliable model for defection prediction, whereas the operation phase is performed periodically (i.e. weekly using a sliding-window method) to detect potential defectors and provide information to guide them in a desirable direction. The final outcomes of the pre-build up phase are rules for frequent and confident trajectories of defectors and non-defectors, which are discovered from customer behavior history of actual defectors and non-defectors. The pre-build up phase consists of four steps. In the first step, all the behavior data related to defection is collected from operational databases and web log files and stored in the CBH (Customer Behavior History) DB. In the next step, SOM is used to determine all the possible behavior states. After determining the behavior states using SOM, all the continuous and complex behavior data in the CBH DB is converted into discrete states. Through this conversion, the current and historic behavior states for each customer are profiled in a TBS (Trajectory of Behavior States) DB as a trajectory which makes it possible to track the dynamics of customer behavior in a new way. In the final step, rules for frequent and confident trajectories of defectors and non-defectors are discovered from the TBS DB and stored in a FCT (Frequent and Confident Trajectories) RB using the association rule mining technique. In the operation phase, potential defectors are detected by comparing their monitored trajectories of behavior states with frequent and confident trajectories of past defectors and non-defectors which were discovered in the pre-build up phase. In the final step, a procedure of defection prevention is performed so as to

lower their likelihood of defection in the next period. Applying a defection detection and prevention procedure periodically makes consistent and continuous implementation of customer relationship management (CRM) possible. The details are explained step by step from this point on.

5.2 Pre-build up phase

5.2.1 Step1: Collecting behavior data and pre-processing

First of all, features which can affect defection are selected based on the prior knowledge of domain experts, and the values of features are collected over the multi-periods from web log files and databases. After collecting behavior data, data preparation is needed to build behavioral profiles. In particular the web log data requires a great amount of effort to be transformed into usage data because of difficulty in session identification and user identification. More detailed information for data preparation of the web log can be found in Cooley et al. (1999). In our online game domain, the totaled session time during a week, the total number of successful sessions during a week, the total number of access errors caused by congestion during a session time, and the average session time are selected by the marketing manager. The values of features are collected through pre-processing and stored in the CBH DB. Table 1 provides behavior data on five customers in the CBH DB. In this table, customer 'AVAL' had spent 1343 minutes playing the game during a week with 23 times of successful access, and he/she had met 9 access errors four weeks earlier. After that, he/she gradually trimmed out his/her usage volume and used only 19 minutes for one week just before his/her final defection.

<Table 1> Sample behavior profiles in the CBH DB

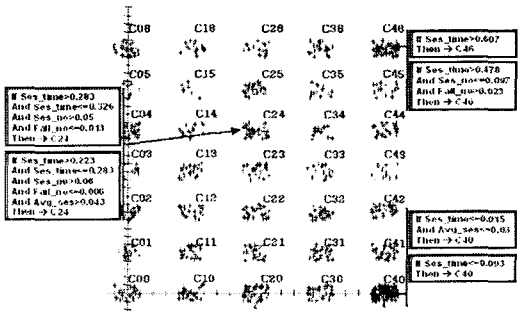
Customer ID	Week	Session Time	Number of normal accesses	Number of error accesses	Average session time	Actual Defection
AVAL	1	19	3	0	6.33	Defected
	2	39	3	0	13.00	
	3	147	5	1	29.40	
	4	1343	23	9	58.39	
DANNYKIM	1	433	18	13	24.06	Defected
	2	2496	15	24	166.40	
	3	8217	29	32	283.34	
	4	3404	25	24	136.16	
ADOON6309	1	10	1	0	10.00	Defected
	2	0	0	0	0	
	3	0	0	0	0	
	4	5	1	3	5.00	
MIO935	1	6457	38	10	169.92	Non-defected
	2	5948	41	50	145.07	
	3	2462	25	21	98.48	
	4	5979	54	36	110.72	
JYOA	1	3748	26	15	144.15	Non-Defected
	2	5318	36	19	147.72	
	3	3933	22	20	178.77	
	4	2575	28	9	91.96	

Normalization is also needed to equalize the contributions of the different variables before feeding them to SOM. Detailed normalization techniques are provided in Chakraborty et al. (2000).

5.2.2 Step 2: Determining the behavior states through SOM training

Based on the selected input features, all the possible states of customer behavior are determined. The input vector for SOM is the value array of the filtered features. In our domain, four behavior features in a past unit period for a certain user consists of an input vector like the record in Table 1. In determining the behavior states using SOM, the total number of states has to be given. We will discuss this issue in a later section. Figure 3 illustrates the possible behavior states which are the results of SOM learning. To interpret each state on the map, decision tree analysis (C5.0) is additionally conducted. All the four behavior features in Table 1 are regarded as independent variables, and the behavior state which has been assigned as a result of SOM learning, is considered as a target class in decision tree analysis. Figure 3 also provides the interpretations for each behavior state on the map.

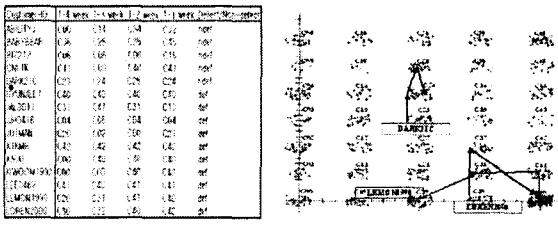
<Figure 3> Behavior states and their interpretations



5.2.3 Step 3: Building the trajectory of behavior states for each customer

After determining the possible behavior states, the history of the behavior state for a certain customer can be visualized as a trajectory on the map, which makes it possible to track the behavior dynamics in a new way. We can make trajectory of behavior states for each customer by evaluating which state the behavior data in each period in the CBH DB belongs to. This process is simply done by running the SOM model which has been built in step 2. Figure 4 shows sample trajectories of behavior states (TBS) and their visualization on the map.

<Figure 4> The sample Trajectories of Behavior States (TBS) and their graphical representation



5.2.4 Step 4: Mining frequent and confident trajectories of defectors and non-defectors

Frequent and confident trajectories of defectors and non-defectors during multi-periods are discovered from the TBS DB, and are used as a scoring indicator for classifying defectors and non-defectors. For this purpose, we apply the Apriori algorithm for association rule mining. The Apriori algorithm discovers frequently co-occurrent patterns that appear often enough to make themselves statistically remarkable. Original the Apriori algorithm ignores the time sequences over time, so we define every items in an itemset as concatenation of attribute and value (e.g. replacing *item* to *period i = item*). Then we can keep the time sequences in discovering frequent and confident trajectories. For the online game domain, we give 3 % minimum support and 60 % minimum confidence as an input for association rule mining, and we obtained 24 negative class rules and 25 positive class rules.

5.3 Operation Phase

5.3.1 Step 1: Monitoring current behavior and updating trajectories

All the current behaviors of customers are collected and converted to a behavior state using the SOM model built in the first phase. Next, trajectories of behavior states in the TBS DB are updated with the new behavior state. Trajectories of behavior states are treated and maintained as a finite length queue. A new monitored behavior state is added to the prior trajectory of those customers and oldest entry is discarded.

5.3.2 Step 2: Detecting potential defectors

Potential defectors are identified by assigning the scores which represent the likelihood of defection. The frequent and confident trajectories of past defectors and non-defectors generated (from training data) in the pre-build up phase are used to score new (or test) data. For this purpose, we adapted the scoring method based on association rules from the works of Liu et al. (2000) and Ma, et al. (2000). Scoring using association rules is a challenging problem because there are often many rules that can be applied when we want to score a data case. They developed a heuristic technique that is reported to be both effective and efficient. Their scoring function uses three types of useful information. The first is *rule confidence*. It is an essential piece of information because it is basically a probability estimate. Thus, confidence plays an important role in the scoring function. The second is *rule support*. Rule support reflects the usefulness of the rule. A rule that covers too few data cases is often overfitted and unreliable. And the third is types of rules. There can be two types of rules that cover a case, positive class rules and negative class rules. This conflict has to be resolved in the scoring function. The scoring function is developed as follows by taking into account the above information (Liu et al. 2000; and Ma et al. 2000).

$$S = \frac{\sum_{i \in POS} W^{i}_{positive} \times conf^{i} + \frac{1}{k} \sum_{j \in NEG} W^{j}_{negative} \times conf^{j}_{positive}}{\sum_{i \in POS} W^{i}_{positive} + \sum_{j \in NEG} W^{j}_{negative}}$$

where,

POS : the set of positive class rules that can cover the data case,

NEG : the set of negative class rules that can cover the data case,

$W^{i}_{positive}$: the weight for the positive class rule i

$$W^{i}_{positive} = conf^{i} \times sup^{i}$$

$W^{j}_{negative}$: the weight for the negative class rule j

$$W^{j}_{negative} = conf^{j} \times sup^{j}$$

$conf^{i}$: the original confidence of the positive class rule

$conf^{j}_{positive}$: the confidence after converting the negative class rule j to a positive class rule, i.e., $conf^{j}_{positive} = 1 - $'the confidence of rule j'.

The value of S is between 0 and 1 inclusively. Customers who have the likelihood of defection greater than a specified threshold can be targeted and defection prevention procedure is activated for them. In our domain, all the customers in the training set and test set are scored. Figure 6 illustrates scoring results for some customers in the training set. For example, the first customer was actually a defector. If we specify the defection threshold as 0.5 then he is classified as a defector because his/her defection score is '0.7772791'.

<Figure 6> Sample customers who are assigned a defection score

Customer_ID	T-4 week	T-3 week	T-2 week	T-1 week	Actual	Score	Predicted(※)
부리뭄	C43	C31	C41	C40	def	0.7772791	def
엘테레아	C23	C22	C20	C40	def	0.6402469	def
르네스	C42	C40	C40	C04	def	0.7446328	def
마젤사데빌	C20	C40	C40	C40	def	0.7914637	def
안두돔샘	C11	C40	C40	C40	def	0.7914637	def
샘봉이시슨	C40	C40	C00	C00	def	0.6778643	def
엘젤어남자델	C40	C40	C40	C40	def	0.7964321	def
미니세니	C13	C20	C21	C31	def	0.1103073	ndef
만루홈런	C13	C04	C22	C23	ndef	0.0568182	ndef
엘진삼중	C13	C03	C03	C14	ndef	0.1250000	ndef
엘진홈살	C06	C40	C01	C06	ndef	0.4835134	ndef
베팅가르온	C30	C40	C40	C03	ndef	0.7446328	def
인플레써엘	C46	C46	C46	C46	ndef	0.0226117	ndef

※ Defection threshold = 0.5

5.3.3 Step 3: Preventing defection

The objective of the defection prevention step is to provide information of To-Be behavior state for next period to drive their potential defectors into a desirable position in terms of their behavior states. If we can find a behavior state (i.e. it corresponds to a neuron on SOM) which will lower the likelihood of defection, then we can design a campaign to focus on modifying their behavior

patterns so that they belong to the desirable state in the next period. In this section, two strategies are suggested to respond to the question of 'Where to drive potential defectors for next period?' according to approach for modification of customer behavior. The first strategy is to adopt the improvement approach in inducing potential defectors to change their behavior pattern. This strategy is based on the fact that it is very difficult (or impossible) to change customer behavior greatly in such a short period (i.e. a week in our example). Thus, the basic principle of this strategy is inducing the gradual change in their behavior pattern through the regular and repetitive operation of this procedure. The second strategy is using the innovative approach in inducing potential defectors to change their behavior pattern. Innovative approach requires high costs and risk in campaign because it tries radical change about customer behavior. In practice, we may have to prevent defection as soon as possible in spite of great money and high risk for the VIP customers who are predicted to defect. Therefore, this strategy pursues inducing potential defectors to change their behavior pattern radically through only a few campaigns. Two strategies can be implemented on the map of SOM using the topology preserving property of SOM. Topology preserving property means the data points lying near each other in the input space will be mapped onto nearby map units on SOM. Therefore, it is relatively easy to drive a customer into a closer state (node) from a current behavior state (node) on the output map of SOM because input behavior patterns between the two states are mostly similar by topology preserving property. In the improvement approach, we do not consider driving the customers to a second or further neighborhood node from current node on SOM at a time. Therefore, we can guide potential defectors to a node which will lower defection score best among zero or the first nearest neighborhood nodes. In the same vein, innovative approach tries to drive potential defectors to a node which will lower defection score best among all nodes on the map of SOM. Two strategies are explained in detail from this point on.

5.3.3.1 Strategy 1: Improvement approach

This strategy tries to drive potential defectors into a behavior state among zero or first nearest neighborhood nodes so as to lower their defection score. The improvement approach can be implemented through the algorithm in Figure 7. This algorithm is performed for each potential defector who has been detected in step 2.

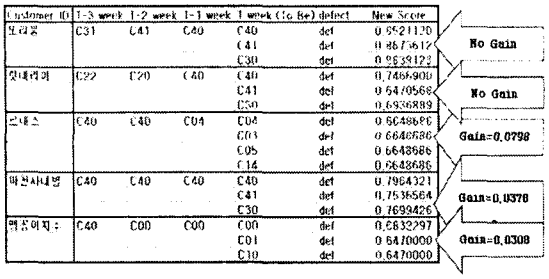
<Figure 7> Defection prevention procedure for improvement approch

```
1  ToBeNode = ∅, MaximumGain = -1
2  0-NeighborNode = {n1}, 1-NeighborNode = {n2, n3, ..., nm}
3  for each neighbor node ni do begin
4    NewTrajectory = add ni to ExistingTrajectory and discard oldest entry
5    scoring the likelihood of defection for NewTrajectory
6    ScoreGain = score of ExistingTrajectory - score of NewTrajectory
7    if ScoreGain > 0
8      if ScoreGain > MaximumGain
9        ToBeNode = ∅, add ni to ToBeNode, MaximumGain = ScoreGain
10     elseif ScoreGain = MaximumGain
11       add ni to ToBeNode, MaximumGain = ScoreGain
12     endif
13   endif
14 endfor
15 recommend to drive to a node in ToBeNode for next period
```

In Figure 7, a trajectory of behavior states (*ExistingTrajectory*) and its defection score are input and final output is a set of To-Be nodes (*ToBeNode*) recommended to drive. At line 1 of the algorithm, an output variable and an internal variable are initialized, and zero and the first nearest neighborhood nodes centered on the current node are selected at line 2. *k-NeighborNode* means the set of *k*th nearest neighbor nodes from the current node. In this notation, $k=0$ means the set of neighbor node which contains only the current node, and $k=1$ means the set of first neighbor nodes which contain four or less neighbor nodes on the rectangular structure SOM. For each selected neighbor node, a new trajectory that is faded by adding the selected node to the existing trajectory is built, and the defection score of new trajectory is evaluated by the scoring function at line 4 and 5. At line 6, a score gain is computed by subtracting the score of a new trajectory from that of an existing trajectory. Among all selected neighbor nodes, the nodes that have maximum and positive score gain are selected through line 7 to 13. Those nodes are recommended as a To-Be behavior states to guide the customer for next period. Thus, the weighting vector of the To-Be node becomes the target behavior pattern of the next period for this customer.

To demonstrate how well the improvement approach performs in our domain, we randomly select 5 potential defectors and compute score gains under the assumption that it can be only driven to zero or the first nearest neighbor node for the next period. Figure 8 illustrates their new scores when they are guided to zero or the first nearest neighbor node in the following week, and their score gains. For two customers, we failed to find zero or the first nearest neighbor node which lowers the defection score in the following week. But with regards to the other three customers, we could lower their likelihood of defection by as much as their score gains. For example, the defection score of the fourth customer was 0.7915 in Figure 6 but if he/she is guided to the 'C41' state in the following period then his/her new defection score will be 0.7537 (See Figure 8). Therefore we can obtain a 0.0378 score gain through our defection prevention procedure. Although the score gain is small, we have a high possibility of defection prevention through the repetitive operation of this procedure.

<Figure 8> Computing the score gains



5.3.3.2 Strategy 2: Innovative approach

Innovative approach tries to drive potential defectors to a node which will lower defection score best among all nodes on the map of SOM. Figure 9 provides algorithm for this strategy. Most of them are same to previous algorithm but innovative approach investigate every node on the map to provide information for To-Be state. If there exist two or more nodes in final *ToBeNode*, a node that is the nearest from current node is selected to recommend as a To-Be state for next period. We recommend to apply innovative approach to high profitable customers who are predicted to defect because it requires high costs and risk in campaign.

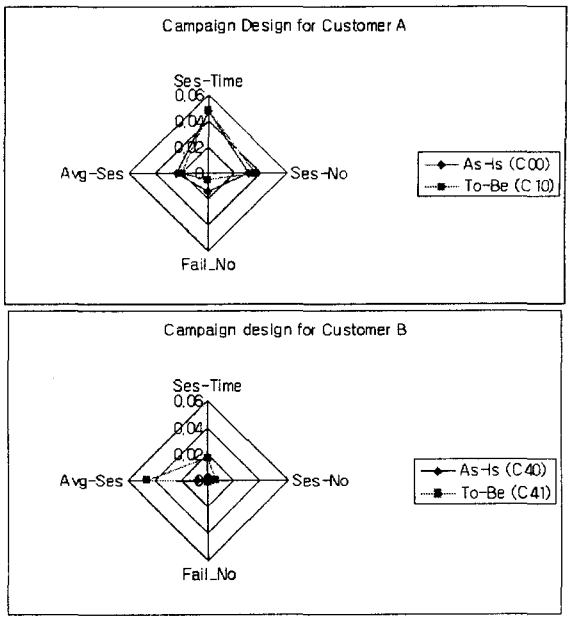
<Figure 9> Defection prevention procedure for innovative approach

```
1  ToBeNode = ∅, MaximumGain = -1
2  for every node n, do begin
3    NewTrajectory = add n, to ExistingTrajectory and discard oldest entry
4    scoring the likelihood of defection for NewTrajectory
5    ScoreGain = score of ExistingTrajectory - score of NewTrajectory
6    if ScoreGain > 0
7      if ScoreGain > MaximumGain
8        ToBeNode = ∅ add n, to ToBeNode, MaximumGain = ScoreGain
9      else if ScoreGain = MaximumGain
10       add n, to ToBeNode, MaximumGain = ScoreGain
11     endif
12   endif
13 endfor
14 recommend to drive to the nearest node from current node ToBeNode for next period
```

5.3.3.3 Personalized campaign design

We can design a personalized campaign based on the information of To-Be state. Figure 10 provides the real case for campaign design. As-is behavior state and To-be behavior state for two customers are presented in this figure. From this figure, the campaign direction for customer 'A' should be reducing the number of access errors caused by congestion. Therefore, it will be the most effective campaign for customer 'A' to recommend non-peak times by mailing a non-peak time discount coupon or sending a manual for access failure. For customer 'B', campaign for extending session length is a prerequisite. Accordingly, it is required to provide customer 'B' additional free access time so as to lower the likelihood of defection.

<Figure 10> Personalized campaign design

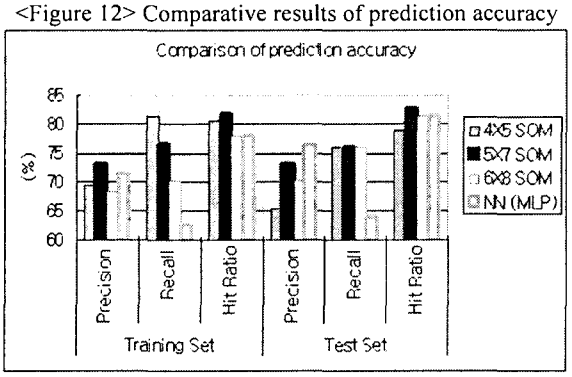


6. Evaluation

This section describes how to improve the prediction accuracy in our approach and compares the results with prediction by the traditional MLP neural network model. In determining the behavior states using SOM, the total number of states has to be given. If we set the total number of states to be very large, then the prediction model will lose the generalization capability because of overfitting. In other words, the final behavior trajectory for a certain user will become very sensitive to a minor change in customer behavior. On the contrary, if we specify the total number of states to be very small, then the model may not have discrimination power between the trajectories of defectors and non-defectors. Therefore, caution is required when making a decision on the number of states. The decision for determining the number of states is usually dependent on the characteristics of the domain and the goals of the decision-makers. The satisfactory number of states may be given by trial and error using past multi-period behavior data and the target variable (i.e. whether they actually defect or not). To evaluate the performance of each number of states, *precision* and *recall* are used as a measure in our procedure. *Precision* (P) is defined as the ratio of the actual number of defectors among predicted defectors to the predicted number of defectors. And, *recall* (R) is defined as the ratio of the predicted number of defectors who are actual defectors to the actual number of defectors in all the training profiles. Precision means what percentage of predicted defectors are actual defectors, and recall means what percentage of actual defectors are detected in our procedure. Precision and recall are computed as $P = \frac{|PD \cap AD|}{|PD|}$, $R = \frac{|PD \cap AD|}{|AD|}$, .where $|A|$ is the number of elements in set A.

To determine the optimal number of states, several number of states are selected by domain experts and the SOM model is trained repeatedly with each number. After training, prediction is performed and the precision and the recall are evaluated for each state number. Finally, the best alternative which has the highest value of precision and recall is selected. Figure 12 shows the comparative results of prediction accuracy among 3 alternatives (i.e. 20 states, 35 states, 48 states) and an MLP neural network (with 4 nodes in hidden layer) in our online game domain. A 5X7 SOM model is most appropriate for this domain because of its high hit ratio, precision and recall. We also tried to predict the possibility of defection using the MLP neural network model based on the same input data. The proposed procedure by the 5X7 SOM model resulted in a slightly higher accuracy than the MLP neural network model. This means that the proposed procedure can cover defection prevention as well as defection detection without deterioration of prediction accuracy compared to that of MLP neural network.

<Figure 12> Comparative results of prediction accuracy



(Unit : %)

Techniques	Training Set			Test Set		
	Precision	Recall	Hit Ratio	Precision	Recall	Hit Ratio
4X5 SOM	69.3	81.3	80.6	65.5	76	78.7
5X7 SOM	73.1	76.6	81.7	73.1	76	82.7
6X8 SOM	68.2	70.3	77.8	70.3	76	81.3
NN (MLP)	71.4	62.5	77.8	76.2	64	81.3

7. Conclusion

We proposed a personalized defection detection and prevention procedure based on the observation that potential defectors have a tendency to take a couple of months or weeks to gradually change their behavior (i.e. trim-out their usage volumes) before their eventual withdrawal. For this purpose, possible states of customer behavior are determined from past behavior data using SOM. Based on this state representation, potential defectors are detected by comparing their monitored trajectories of behavior states with frequent and confident trajectories of past defectors and non-defectors. Also, the proposed procedure is extended to defection prevention for potential defectors which assist building a personalized campaign plan by recommending the desirable behavior state for the next period so as to lower the likelihood of defection. Our proposed approach is a personalized implementation procedure of CRM because it predicts potential defectors and provides a personalized campaign plan for each potential defector through the defection prevention procedure. As an area for further research, we have a plan to develop a system for defection detection and prevention based on our suggested procedure. In this study, we applied our procedure to the online game industry, but it will be a promising research area to apply it to other service industries such as telecommunications, internet access services, and contents providing services, and check the effectiveness of the proposed procedure.

References

Agrawal, R., Imielinski, T. & Swami, A. (1993). Mining association rules between sets of items in large databases. *Proceedings of the ACM SIGMOD Conference on Management of Data*, 207-216.

Agrawal, R. & Srikant, R. (1994). Fast algorithm for mining association rules. *Proceedings of the International Conference on Very Large Databases (VLDB-94)*, 487-499

Alhoniemi, E., Hollmen, J., Simula, O. & Vesanto, J. (1999). Process monitoring and modeling using the Self-Organizing Map. *Integrated CAE* 6(1): 3-14.

Berson, A., Smith, S. & Thearling, K. (2000). *Building data mining applications for CRM*. McGraw-Hill. 277-298

Chakraborty, G. & Chakraborty, B. (2000). A novel normalization technique for unsupervised learning in ANN *IEEE transactions on neural networks* 11(1): 253-257.

Cooley, R., Mobasher, B. & Srivastava, J. (1999). Data preparation for mining world wide web browsing patterns. *Knowledge and information systems* 1(1): 5-32.

Datta, P., Masand, B., Mani, D. R. & Li, B. (2000). Automated cellular modeling and prediction on a large scale. *Artificial Intelligence Review* 14: 485-502.

Eiben, A. E., Koudijs, A. E. & Slisser F. (1998). Genetic modelling of customer retention. *Proceedings of First European Workshop on Genetic Programming*, 178-186.

Kasslin, M., Kangas, J. & Simula, O. (1992). Process state monitoring using Self-organizing maps. *Artificial Neural Networks* 2: 1531-1534.

Kohonen, T. (1990). The Self-Organizing Map.

Proceedings of the IEEE 78(9): 1464-1480.

Kohonen, T. (1995). *Self-Organizing and Associative Memory*. Berlin: Springer-Verlag.

Kohonen, T., Oja, E., Simula, O., Visa, A. & Kangas, J. (1996). Engineering applications of the Self-Organizing Map. *Proceedings of the IEEE* 84(10): 1358-1384.

Liu, B., Ma, Y., Wong, C. K. & Yu, P. (2000). *Target selection via scoring using association rules*. IBM Research Report 21697, NY.

Ma, Y., Liu, B., Wong, C. K., Yu, P. S. & Lee, S. M. (2000). Targeting the right students using data mining, *Proceedings of 6th international conference on KDD 2000*, 457-464.

Mozer, M. C., Wolniewicz, R., Grimes, D. B., Johnson, E. & Kaushansky, H. (2000). Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on neural networks* 11(3): 690-696.

Ng, K. & Liu, H. (2000). Customer retention via data mining. *Artificial Intelligence Review* 14(6): 569-590.

Raghavan, N., Bell, R. M. & Schonlau, M. (2000). Defection Detection : Using online activity profiles to predict ISP customer vulnerability. *Proceedings of the sixth International Conference of Knowledge Discovery and Data Mining*, 506-515.

Simula, O., Vasara, P., Vesanto, J. & Helminen, R. (1999). *The Self-Organizing Map in industry analysis*. CRC Press.

Simula, O., Vesanto, J., Alhoniemi, E. & Hollmen, J. (1999). *Neuro-Fuzzy Techniques for Intelligent Information Systems*. Springer Verlag.

Smith, K. A., Willis, R. J. & Brooks, M. (2000). An analysis of customer retention and insurance claim patterns using data mining : a case study. *Journal of the operational research society* 51: 532-541.

Song, H. S., Kim, J. K. & Kim, S. H. (2001). Mining the change of customer behavior in an internet shopping mall, *Expert systems with applications* 21(3): 157-168.

Tryba, V. & Goser, K. (1991). Self-organizing feature maps for process control in chemistry. *Proceedings of international conference on artificial neural networks (ICANN 91)*, 847-852.

Vesanto, J. (1999). SOM-based data visualization methods. *Intelligent data analysis* 3: 111-126.

Yeo, A. C., Smith, K. A., Willis, R. J. & Brooks, M. (2001). Modelling the effect of premium changes on motor insurance customer retention rates using neural networks. *Lecture notes in computer science* 2074: 390-399. Springer-Verlag.