

Customer Churn Prediction: A Cognitive Approach

Damith Senanayake, Lakmal Muthugama, Laksheen Mendis, Tiroshan Madushanka

Abstract—Customer churn prediction is one of the most useful areas of study in customer analytics. Due to the enormous amount of data available for such predictions, machine learning and data mining have been heavily used in this domain. There exist many machine learning algorithms directly applicable for the problem of customer churn prediction, and here, we attempt to experiment on a novel approach by using a cognitive learning based technique in an attempt to improve the results obtained by using a combination of supervised learning methods, with cognitive unsupervised learning methods.

Keywords—Growing Self Organizing Maps, Kernel Methods, Churn Prediction.

I. INTRODUCTION

CUSTOMER Relations Management is a prominent field in the business analysis process of any organization. It deals with both identifying, expanding to, attracting potential customers, as well as retaining existing customers. Customer Relations Management has two major aspects, one being the operational aspect and the other being the technical aspect. This technical aspect is also known as Customer Analytics. Customer Analytics can be broken into two major categories.

- 1) Descriptive Analytics : Customer Identification
- 2) Predictive Analytics : Customer Retention

In this research, we have focused on predictive analytics and attempted to obtain a considerable improvement in this. First, one must consider the domain, and the following sections will give an overview of this.

A. Customer Churn Prediction

Being of the Predictive Analytics category, customer churn analysis is mainly focusing on retaining customers. In the competitive business environment of modern markets, it is not rare to see customers defect from a producer's product. This may be over a competitor's product or by opting for a new range of products altogether (analogous to abandoning film-reels altogether for Digital Cameras). The probability (or in some cases, the certainty) that an existing customer will defect is known as 'churn' and in churn analysis, an attempt to identify those customers who are likely to churn or defect is made.

The typical approach is by analyzing the existing records of instances where the customers have already churned, and using data analytics techniques to identify the potential defections of

existing customers. These analytics techniques could be either statistical models or computerized algorithms.

However, due to the advances of data collection techniques and electronic storage of transactional data, the volume of data that needs to be analyzed is immense and presents a daunting task for the data analyst. Therefore, the automation of the analytics task is called for, and therefore, relatively new techniques must be used for this.

B. Machine Learning & Data Mining

Machine Learning, and Data Mining techniques are used to automate the analytics tasks that are too time-consuming or labor-intensive for human analysts to conduct manually. This is a popular field of study in computer science as it has revolutionized the way data analytics is done. The practical applicability of these techniques not only cover the customer analytics domain but various other disciplines such as astronomy, bio-medical sciences, ecology, economics, document classification, and demographic and social studies as well. The tasks these techniques are used for can differ from Natural Language Processing, Epidemiology, Stock Market Prediction, the Human Genome Project and Molecular Structure Prediction.

Machine learning, again can be identified in two main areas;

- 1) Supervised Learning :

Here the goal is to build a predictive model based on observations. Given a data manifold, containing the observed response variable values, the task is to build a model that will predict a significantly accurate response variable for a data point, of which, the response is not known a priori. This is hence, a predictive analysis task. This can be a classification into separate discrete classes, or a regression to a continuous region.

- 2) Unsupervised Learning :

As opposed to supervised learning, the response variable, in this case, is not specified. Instead, the goal is to identify an underlying pattern of the data. Typically, when a data set is provided, the goal is to identify a clustering, in which within a cluster the variability is minimal, and between clusters, the said variability is maximal. An example, in the customer domain, will be an analysis which identifies the segments within the market based on various data fields, be them RFM (Recency, Frequency and Monetary parameters) or demographic data.

In this research, the major focus is the task of churn analysis, which is a predictive analysis, which in turn implies that a predictive approach by using supervised learning can be used. However, unsupervised learning shall also be used to improve the obtained results.

Damith Senanayake is a student at Department of Computer Science and Engineering, University of Moratuwa, Sri Lanka, 10400: (e-mail damith.10@cse.mrt.ac.lk).

Tiroshan Madhushanka, Laksheen Mendis and Lakmal Muthugama are with the Department of Computer Science and Engineering, University of Moratuwa.

The following sections will, therefore, discuss some related work existing in the respective disciplines involved, and our approach. Later the obtained results will be discussed.

II. RELATED WORK

A. Use of Data Mining in Customer Data Analytics

In the field of customer analytics, various features are involved as variables. There are several categories of them and most noteworthy will be the customer variables of recency, frequency and monetary value (referred to as RFM), and demographic variables such as geographical information, cultural information (ethnicity, religion, gender, etc.) and age [1]. These variables can be categorized as follows;

- 1) Transaction features: otherwise known as the RFM captures the recency, frequency and monetary value of a given transaction. Most of the times, this category is the more highlighted set of features.
- 2) Demographic features: the information about the customer themselves, such as geographical, ethnic, religious, income level, occupational, gender information. These also can be taken into account to make the mining more relevant.

Customer Analytics encompasses the discipline of Customer Identification through their purchase habits and transactional data, which can be vital for many fields. Marketing, advertising and offer extension are such fields where an insight of the customer segmentation will be extremely useful.

As for the focus of this paper, the prevention of customer churn through customer retention is a key issue of CRM. The possibility of customer churn and importance of customer retention can be easily understood based on following reasons [2].

- 1) Some customers are cherry pickers (they visit multiple places to shop the basket of goods)
- 2) Some customers are store switchers
- 3) Some uses few retailers for their shopping
- 4) Retaining existing customers is advantageous than acquiring new ones
- 5) Loyal customers are advantageous as they increase spending amount(upselling and cross selling), spread positive word-of-mouth, less costly to serve, less prone to noise by competitors.

Because of the mentioned reasons there exist many machine learning approaches majorly focusing on customer Retention Analysis or Customer churn analysis .One-to-one marketing, loyalty programs and complaint management are classified as key elements in customer retention [3].

But there is no clear consensus about the relationship between customer retention and profitability because the most loyal customers are not necessarily the most profitable ones in the business domain. Therefore it is important to investigate both customer retention and profitability outcomes, while

explicitly testing for differences with respect to the impact of the same set of explanatory variables on both outcomes [4].

Most of the studies in customer retention have focused on increasing the accuracy of predicting churn using the data mining techniques. Just a simple predicting churn can not reduce its rate of occurrence since the churn can be a result of a number of factors [5]. Therefore behavioral data of customers plays a significant role and they can be used to evaluate customers potential value [6], can be used to assess the risk they stop paying their bills, and predict their future needs [7].

B. Self Organizing Maps and variants

Originally proposed by Teuvo Kohonen [8], the self organizing map is a neural model inspired by the organization of neurons in the human brain. Some areas of the human brain are more attuned to specific tasks than others [9], for instance, the olfactory lobes of the cortex is more specialized in identifying smells and flavors. A similar model is used in self-organizing maps to obtain a *specialty on a locality* of a dataset on top of the map or the neural network. This is obtained by exploiting the principle of vector quantization [10] which is used in signal processing to encode a manifold into a submanifold, with little or no loss of topological information of the original manifold. The approach is to keep a set of codebook, or reference vectors. These codebook vectors (typically) are less in number as compared to the data manifold, but have the same dimensionality. Formally, this could be denoted as a set $C \in \mathbb{R}^d$, of size m where $d \in \mathbb{R}$ is the dimensionality of the original data $V \in \mathbb{R}^d$ of size n .

In self-organizing maps, this concept is used as a base and the *reference vectors* are represented by a grid (usually of two dimensions) of neurons, whose input weights bear the same dimensionality as the input data. The voronoi tessellation and error optimization is done in two steps, the mathematical derivations are defined as;

- Assignment

An input is assigned to a neuron, which is called the winner node or the best-matching unit (bmu) which satisfies the following condition:

$$bmu = \operatorname{argmin}_{neuron \in Map} \|neuron.weight - input\|$$

- Adjustment

Coming directly from the optimal solution for the quantization error, use the following equation to adjust the weights of the neurons of the map,

$$w_i(t+1) = w_i(t) + \alpha(t)h(bmu, n_i)[n_i - bmu]$$

where w_i is the weight vector of the neuron i on the map, h is the neighborhood function, and α is the learning rate.

It is noteworthy that the norm considered here is strictly the euclidean norm. The above algorithm results in a grid of

neurons which preserves the topology of the original data. This is a useful trick in preprocessing as it could retain the underlying structure while reducing the dimensionality to a small number (typically only two). However, as is evident, there exist many issues with this algorithm, such as the fact that the size of the map needs to be predefined by the user, and there is no guarantee that the euclidean norm will adequately cater to the need of the data domain.

The Growing Self Organizing Map or the GSOM is an attempt to overcome the need to define the size of the map *a priori*. The approach [11] is to add a growth capability with a mechanism to control the growth. This algorithm relies on the discrepancy between a best matching unit and the relevant input vector. Therefore each neuron contains an associated error for it. Everytime a neuron emerges to be the winner node for an input, this error is updated with respect to the said discrepancy. This is denoted by

$$E_i(t+1) = E_i(t) + \sqrt{|w_i - x|^2} \quad (1)$$

In this original paper, the authors have indicated to use the euclidean distance for the norm considered in the above equation.

As the error is updated, once the residual error of a given neuron exceeds a parameter, dubbed the Growth Threshold, the error is then distributed to its neighbors. However, in case the best matching unit is located at a boundary, and there exists no neighboring node in the direction to grow, a new node is created. Also, it is note worthy, that since the initiation of the weight vectors of the new neurons is so that they are suited for the neighboring neurons, which in turn are now adapted to the input, the growth asymptotically stabilizes.

C. Kernel Methods

It is obvious that the above algorithms make use of the concepts of Euclidean norms and Euclidean distances. However, some domains may yield data that may not operate well within a Euclidean space. An example would be, in a scenario of customer analysis, when the day of the week a certain purchase is made, is taken into consideration. In such a way, any linear encoding of the weekdays, would not encapsulate the semantics such that a week that begins from Sunday, will be more similar to the end of the week - i.e. Saturday than the middle of the week.

Kernel methods use a phenomenon called the *kernel trick* to overcome this issue. Here, the data is mapped into a (typically of higher dimensions) Hilbert space, in which, the Euclidean properties may hold. A large amount of literature is found on Kernel methods, but we shall only briefly discuss the core concepts of it.

Kernel methods are instance based learners which are used to pattern recognition and map data into higher dimension spaces in order to data to be more separated using a mapping function. In kernel methods, kernel trick mathematical tool is used to avoid the explicit mapping of candidate linear algorithms with a nonlinear algorithms and can be expressed as a inner product of mapping function $\phi(x)$,

i.e $K(x, y) = \langle \phi(x), \phi(y) \rangle$ where $\phi : X \rightarrow F$, F is a high dimensional inner product feature space.

The function $K : X \times Y \rightarrow \mathbb{R}$ is referred as a kernel or kernel function. Since kernels are used the mapping function ϕ does not need to be explicitly computed.

1) *Kernel Self Organizing Map*: Graepel et al Initially proposed a kernelized version of SOM that focused on optimizing the topographic mapping [12]. Then introduced a Kernel SOM which applies to the images of input data using a mapping function. [13], [14] .

- Batch Kernel SOM

This Kernel SOM is based on K-means and each data point x mapped to feature space using the mapping $\phi(x)$. For this each mean is presented as a weighted sum of the mapping functions, $m_i = \sum_n \gamma_{i,n} \phi(x_n)$ and then selects a mean with the minimum distance between the mapped point and the mean. The details of this algorithm can be found on the original paper by Mac Donald et al [14].

However, it is noteworthy that the whole set of training data must be provided to train a map, hence online learning is not a possibility with this kind of maps.

- Kernel Dissimilarity SOM

In this approach, first data points are mapped to the feature space and then apply the SOM in the mapped space. The wining rules have proposed in two ways as follows.

As an input space,

$$v = \underset{i}{\operatorname{argmin}} \|x - m_i\|$$

As a feature space,

$$v = \underset{i}{\operatorname{argmin}} \|\phi(x) - \phi(m_i)\|$$

The weight updating rule,

$$m_i(t+1) = m_i(t) + \alpha(t)h(v(x), i)\nabla J(x, m_i)$$

where $J(x, m_i)$ is the distance function in the feature space as follows,

$$\begin{aligned} J(x, m_i) &= \|\phi(x) - \phi(m_i)\|^2 \\ &= K(x, x) - 2K(x, m_i) + K(m_i, m_i) \end{aligned}$$

then,

$$\nabla J(x, m_i) = \frac{\partial K(m_i, m_i)}{\partial m_i} - 2 \frac{\partial K(x, m_i)}{\partial m_i}$$

and $\alpha(t)$ and $h(v(x), i)$ are the learning rate and neighbourhood function respectively.

III. OUR APPROACH

For experimentation with the methodology that we have suggested in this paper, we have used a churn data set provided by www.KNIME.org as a benchmark. The dataset contains 21 fields, three of which are response fields (*churn*, *VMail Plan*, and *Int'l Plan*), and one an ID field (*Phone*). It contains 3333

entries and other fields contain information such as the length of calls, number of calls, the service options purchased etc.

The workflow we conducted in running a churn analysis here is noted below.

A. Data Preprocessing

The goal here was to test the use of the Kernelized version of the GSOM (described in Data Mining) against other methods without the use of which. Therefore, rather than attempting to have an optimal score by a strenuous flow of preprocessing, we have attempted to compare the KGSOM algorithm against a minimally preprocessed dataset. Therefore, the steps followed are as below.

1) Data Reduction

Initially, the response fields, along with the ID field were dropped from the analysis. However, the target response variable - churn - was saved for training the regression / classification process.

Next, two methods were followed to extract the required features. First one was to conduct χ^2 -tests on each of the manifest variables, and find the best k of them. Next was to conduct a principal component analysis (PCA) to extract the k features. It was found that these could be used interchangeably, arguably due to the nature of this particular dataset.

Next, random sampling was done to separate a training and test dataset, with according proportions.

2) Data Encoding

The data contained several fields of categorical data. These were encoded into numbers. However, bitmap indexing was not used as one of the goals here was to assess the possibility of forgoing the use of bitmap indexing to compensate for the loss of accuracy when dealing with encoded categorical data, and gain considerably good results.

B. Data Mining

The process of mining the dataset essentially meant training a supervised learning model on the prepared dataset. This could either be a classifier or a regressor. Without loss of generality, we have used a regression model for the task.

However, prior to training the model on the single dataset, a naive ensemble approach is followed here. That is by clustering the dataset to achieve a high homogeneity within the clusters and a high heterogeneity between clusters. This was done by using a kernelized version of the Growing Self Organizing Map (or KGSOM).

1) The Kernel GSOM

Building atop the work of Alahakoon *et al.* [11] and Mac Donald and Fyfe, we developed a version of the growing self organizing map which incorporated kernel methods into it. We have done this by defining a kernelized growth threshold as

$$GT_k = GT \times \eta_k \quad (2)$$

Where η_k is a function based on the kernel we are using. For instance, in our case, as we used the Gaussian Kernel, we used $\eta_k = \frac{1}{\sigma^2}$ for the experiments.

The final algorithm can be summed up as shown in Algorithm 1.

Algorithm 1 kernel GSOM

```

1: procedure KGSOM
2:   create map  $M$  with four initial nodes with random weights
3:   for input  $x$  in training set do
4:      $f(x) \leftarrow \operatorname{argmin}_{r \in M} \{-2k(x, m_r)\}$ 
5:      $M_{f(x)}.Err(t+1) \leftarrow M_{f(x)}.Err(t) + \frac{M_{f(x)}.Err(t) + 1}{\sqrt{k(x, x) + k(M_{f(x)}, M_{f(x)})} - 2k(x, M_{f(x)})}$ 
6:     if  $M_{f(x)}.Err \geq GT_k$  then
7:        $M_{f(x)}.Err \leftarrow 0.5 \times GT$ 
8:       for each neighbor  $n$  of  $M_{f(x)}$  do
9:          $Err_n(t+1) \leftarrow Err_n(t) + \gamma Err_n(t)$ 
10:      if  $n$  not found then create new node  $n$ 
11:   for each node  $m \in M$  do  $w_m = w_m + \alpha(t)h(m, f(x))\nabla J(x, w_m)$ 

```

2) *Pruning the KGSOM*: Further improvements were made to this algorithm to improve the speed of execution and convergence to a clustering. One major noteworthy trick was to prune the map during the training.

There were multiple options while pruning the map.

One was to train the whole map, and prune at the end, which would be an obsolete attempt to increase the speed of execution. The other, was to intermittent pruning while the algorithm was being executed.

Therefore, pruning was done after every k th iteration. This pruning was done under two criteria.

- 1) The age of the node : it was observed that the older nodes if deleted, could output more separation of the clusters. Therefore, we have considered a generation threshold, the older nodes than which were deleted. This threshold was a function of the iteration i . For instance, a simple threshold function $PT_g(i)$ (the Pruning Threshold based on Generation at the i th iteration) could be defined as :

$$PT_g(i) = \frac{i \times C_\pi}{l} \quad (3)$$

where i is the current iteration, l is the maximum number of iterations, and C_π is the pruning coefficient.

- 2) Usage of the node : Furthermore, to reduce the variability of outliers, we pruned away the nodes that were used below a usage threshold, which was similarly defined as:

$$PT_u = PT_g \times n \quad (4)$$

where n is the number of inputs.

Note that this however, only works given that the training is a batch training.

With these improvements, we were able to achieve a significant increase in speed of convergence to the final clustering. In average, where the number of nodes in the clustering converged to stable configuration around 600 nodes and took about 85 iterations for convergence in the unpruned KGSOM, we were able to reduce it to 250 nodes and

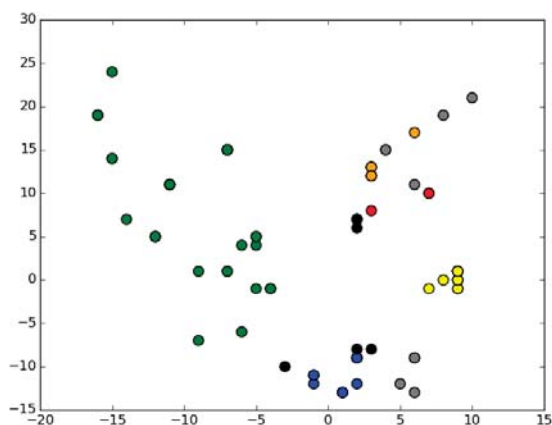


Fig. 1: Clustering of the Zoo dataset using the GSOM

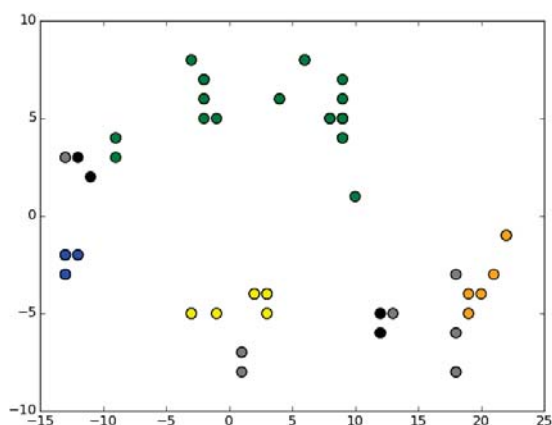


Fig. 2: Clustering of the Zoo dataset using the KGSOM

45 iterations in average. these savings were considerably advantageous in achieving a better clustering and faster results.

After identifying clusters, a separate model was trained on each of the clusters. Therefore, the prediction process also was done in two steps;

- 1) Feed the data entry that needs a prediction into the GSOM to identify the neuron which matches it best
- 2) Use the model trained on the cluster which contains the said neuron to obtain the predicted value of the churn

IV. RESULTS AND VISUALIZATIONS

The initial experiments were run to obtain the results of the Kernel GSOM. This produced comparable results with the existing version of the GSOM. Fig. 1 and Fig. 2 demonstrate the trained map on the UCI zoo dataset, which is a standard in testing the GSOM. And Table I encapsulates the analysis of the clusters. As this was done at an earlier stage of our development, we were not attempting to perfect the scores of these results. However, the average of using various parameter sets, we obtained the following results.

TABLE I ACCURACY MEASURE COMPARISON BETWEEN GROWING SELF-ORGANIZING MAP AND THE GROWING KERNEL SELF-ORGANIZING MAP

	Average Precision	Average Recall	Average F_1
GSOM	0.8894	0.4250	0.4858
GKSOM	0.9182	0.7036	0.6705

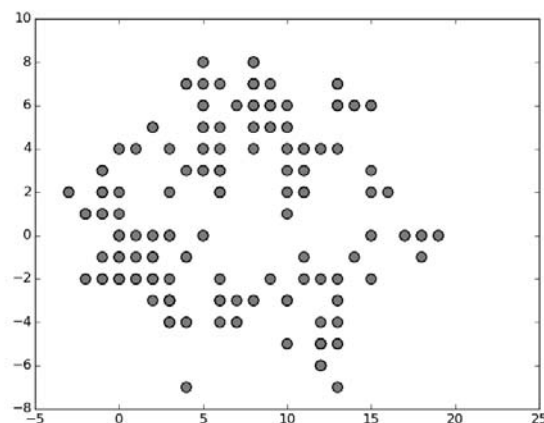


Fig. 3: Clustering the Churn Data with GSOM

The churn data, also showed some empirical differences of clustering as it was fed to the two different versions of the GSOM. And we noted that using the KGSOM we could obtain better cluster separations.

The Fig. 3 and Fig. 4 display the clusterings obtained by the GSOM and KGSOM respectively.

The clusterings, that were fed into the regression algorithms were visualized in the two-dimensional space as below, where blue represents nodes with heavy churn and orange otherwise.

However, as the main focus of this research was to analyze the churn prediction. The results that we obtained are as below.

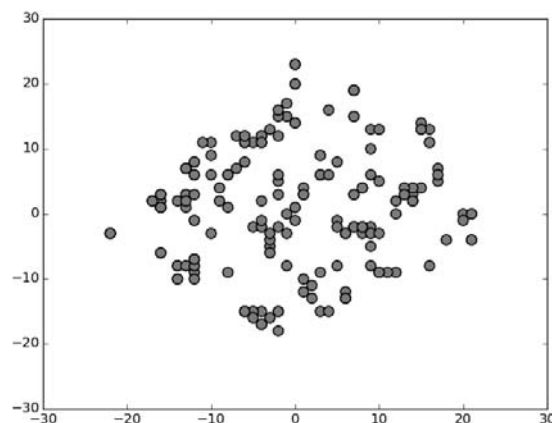


Fig. 4: Clustering the Churn Data with KGSOM

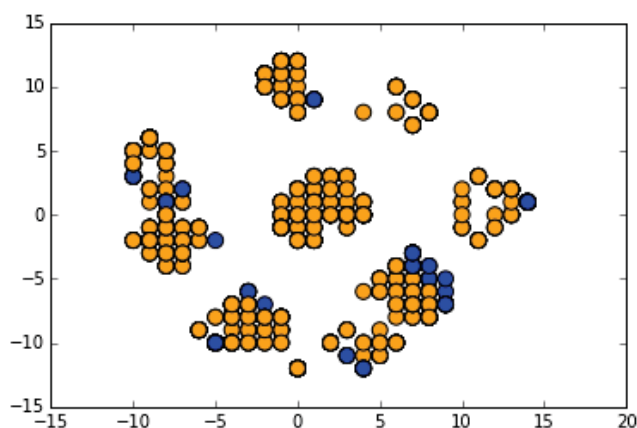


Fig. 5: KGSOM clustering as preprocessing

A. Comparison between with and without Using Kernel GSOM

From the set of experiments run on the churn dataset using both entropy-based models such as support vector machines and random forests, and the generalized linear models such as logistic regression, we observed that the KGSOM preprocessing had a significant impact on the results. The results were taken at different parameter sets for the KGSOM, as well as the number of fields involved were considered.

It was observed that in the case of this particular dataset, using 10 of the features involved resulted in the optimal results. Therefore, the best 10 features out of the χ^2 - test were fed into the algorithm.

The KGSOM parameters were changed accordingly. Initially, the variance parameters were set manually, to see if it could be used to improve the results, but it was noted that automating the variance parameter of the dataset (σ^2) provided better results.

The other parameters that we changed to correlate with the results were the pruning coefficient of the map, and the spread-factor of the map, which both directly and indirectly affected the number of nodes in the final map, and therefore the quality of the clustering. We noted that a best score was usually found where the pruning coefficient and the spread-factor were both around 0.8.

Finally, the area under the receiver operator curve in the cross-validation was used to obtain the comparative results of the churn analysis.

1) *For Entropy Based Methods:* For entropy based methods, the benchmark score, without bitmap-indexing or using the KGSOM was in the vicinity of 0.78. Given that the data was relatively unbalanced (only about one of seven customers had defected in the data) this score was significantly accurate, especially without much preprocessing.

However, using the KGSOM as a preprocessing technique reduced accuracy to 0.74 at best. The reason behind this, may be the fact that by clustering the original data, we reduce the number of entries to be fed into the algorithms, which reduces the model accuracy.

TABLE II: THE ROC-AUC SCORES OF THE EXPERIMENT INSTANCES WITH RESPECTIVE ALGORITHMS

Algorithms	Benchmark	Final Score
GLM	0.52	0.71
Entropy-based	0.78	0.74

2) *For Generalized Linear Models:* For generalized linear models, the benchmark score without using bitmap-indexing or other preprocessing techniques was around 0.52. The lowness of the score, is due to the non-euclidean nature of data which makes the encoding meaningless in the real-valued dataspace, which makes the model inaccurate.

Using the KGSOM as a preprocessing technique, this accuracy could be improved to 0.71. Which is an increase of nearly 40%. This is due to the fact that the clustering provided by the KGSOM uses the kernel trick into disregarding the euclidean similarities in the encoded dataspace.

These results could be summarized as below

V. CONCLUSIONS

As suggested by the evidence provided through the experiments, it is obvious that the tools of machine learning are quite formidable in analyzing customer churn. However, more specifically through this research we were able to determine the usability of relatively unused techniques and to obtain an idea about the practical implications of the empirical results.

A. Strengths and Limitations

It was noted through the experimental and empirical evidence that using the kernel based clustering method as a preprocessing technique, the accuracy of generalized linear models could be improved significantly. However, the reduction of samples in training sets caused the results of entropy-based methods to degrade slightly.

This implies that for generalized linear models, where encoding of categorical data ensues semantic issues of the value encoding in practice, the use of the kernalized GSOM will help alleviating the reliance of euclidean distance measures, while improving the regressor accuracy.

B. Practical Implications

The practical implications of these results are broad, however, the major implication is that the use of kernalized clustering methods in general, could potentially be a replacement for preprocessing techniques or tricks that we use to deviate from euclidean assumptions. For instance, one of the major techniques that we use, is bitmap indexing, which is problematic in categorical variables with a large cardinality in the domain (aptly dubbed, 'the curse of cardinality'). For implementations of supervised learning algorithms, kernalized clustering could provide a shortcut to eliminating the reliance on euclidean measures.

However, it must also be noted this generalization is only valid for instances where modeling of regressors are done in a mathematical regression model and not a bayesian

or entropy-based platform, i.e. these improvements can be significantly seen in methods such as Generalized Linear Models, but not in Support Vector Machines or Decision Trees.

ACKNOWLEDGMENT

The authors would like to thank Dr. Sumith Matharage, Industrial Lecturer at the School of Information and Business Analytics of Deakin University, Melbourne, Australia, Dr. Shehan Perera, Senior Lecturer at the University of Moratuwa, Sri Lanka, and Professor Damminda Alahakoon, Associate Professor, School of Information and Business Analytics of Deakin University for the guidance and supervision provided in this research. Also, the authors thank and acknowledge all the researchers whose work has laid the foundation to this work.

REFERENCES

- [1] K. Tsipis and A. Chorianopoulos, *Data mining techniques in CRM: inside customer segmentation*. John Wiley & Sons, 2011.
- [2] V. L. Miguéis, D. Van den Poel, A. S. Camanho, and J. Falcão e Cunha, "Modeling partial customer churn: On the value of first product-category purchase sequences," *Expert systems with applications*, vol. 39, no. 12, pp. 11 250–11 256, 2012.
- [3] E. W. Ngai, L. Xiu, and D. C. Chau, "Application of data mining techniques in customer relationship management: A literature review and classification," *Expert systems with applications*, vol. 36, no. 2, pp. 2592–2602, 2009.
- [4] B. Larivière and D. Van den Poel, "Predicting customer retention and profitability by using random forests and regression forests techniques," *Expert Systems with Applications*, vol. 29, no. 2, pp. 472–484, 2005.
- [5] B.-H. Chu, M.-S. Tsai, and C.-S. Ho, "Toward a hybrid data mining model for customer retention," *Knowledge-Based Systems*, vol. 20, no. 8, pp. 703–718, 2007.
- [6] T. C. Hung, C., "Segmentation based on hierarchical self-organizing map for markets of multimedia on demand," 2008.
- [7] L. G. S. Berry, M. J. A., "Data mining techniques: For marketing, sales, and customer support." 2003.
- [8] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.
- [9] P. Rakic, "Specification of cerebral cortical areas," *Science*, vol. 241, no. 4862, pp. 170–176, 1988.
- [10] R. M. Gray, "Vector quantization," *ASSP Magazine, IEEE*, vol. 1, no. 2, pp. 4–29, 1984.
- [11] D. Alahakoon, S. Halgamuge, and B. Srinivasan, "Dynamic self-organizing maps with controlled growth for knowledge discovery," *Neural Networks, IEEE Transactions on*, vol. 11, no. 3, pp. 601–614, 2000.
- [12] M. B. T. Graepel and K. Obermayer, "Self-organizing maps: generalizations and new optimization techniques," *Neurocomputing*, 1998.
- [13] P. Andras, "Kernel-kohonen networks," *International Journal of Neural Systems*, 2002.
- [14] C. F. D. Mac Donald, "The kernel self organizing map," in *Proceedings of 4th International Conference on knowledge-based intelligence engineering systems and applied technologies*, 2000.

Damith Senanayake is a final year undergraduate at the University of Moratuwa, Department of Computer Science and Engineering, finishing his reading for the BSc(Hons), Eng in 2015. Born in the vicinity of Colombo, his areas of interest encompass bioinformatics/ computational biology, machine learning, data mining, information retrieval and multivariate statistics.

Lakmal Muthugama is a final year undergraduate at the University of Moratuwa, Department of Computer Science and Engineering, finishing his reading for the BSc(Hons), Eng in 2015. Born in Colombo, he is an individual keen on researches in the areas of Geographical Information Systems, machine learning, data mining and information retrieval.

Laksheen Mendis is a final year undergraduate at the University of Moratuwa, Department of Computer Science and Engineering, finishing her reading for the BSc(Hons), Eng in 2015. Born in Colombo, she is proficient in the fields of management, economics, business analysis, financial accounting and her areas of interest are business intelligence, customer relations management, data mining and machine learning.

Tiroshan Madushankha is a final year undergraduate at the University of Moratuwa, Department of Computer Science and Engineering, finishing his reading for the BSc(Hons), Eng in 2015. Born in Kaluthara, he is an enthusiastic software engineer, with research interests of business intelligence, data mining and machine learning, artificial intelligence and data visualizations.