PREDICTING CHURN IN MOBILE TELECOMMUNICATIONS INDUSTRY

Ionut B. BRANDUSOIU Gavril TODEREAN
Technical University of Cluj-Napoca
ionut.brandusoiu@gmail.com, toderean@pro3soft.ro

Abstract: In today's competitive market, the customer retention represents one of the focal concern for telecommunications companies. The telecommunications sector can be considered on the top of the list with an annual churn rate of approximately 30%. Since the cost associated with subscriber acquisition is higher than the cost of subscriber retention, the need for predictive models to identify subscribers that are at risk of churning has become an important aspect for modern telecommunications operators. But, because the pre-paid mobile telephony market is not easily traceable and definable, implementing a predictive model would be a complex process. In this paper, we present an advanced methodology that helps to predict subscribers churn in pre-paid mobile telecommunications industry, by applying data mining algorithms on dataset that contains call details records. To construct the models we will use k-nearest neighbor, logistic regression, and bayesian networks algorithms. To evaluate and compare the models, we use the gain measure.

Keywords: *bayesian networks, call details records, churn prediction, k-nearest neighbor, logistic regression, telecommunications.*

I. INTRODUCTION

Customer retention is considered to be one of the main concerns of companies in the telecommunications industry. With the increasing competition and diversity of offerings on the market, many telecommunications companies take advantage of data mining techniques to predict customer churn [1].

In the mobile telecommunications industry, the churn term, also known as customer attrition or subscriber churning, refers to the phenomenon of loss of a customer [2]. It is measured by the rate of churn and is an important indicator for organizations. This process of movement from one provider to another is usually happening due to better rates or services, or due to different benefits that a competitor company offers when signing up.

In an almost saturated market, companies are using a defensive marketing strategy to keep their existing customers. In order to accomplish this, they need a method that can identify the customers who are most likely to churn, so that they can deploy proactive retention campaigns [3]. To maximize effectiveness and to reduce the high cost involved in these retention campaigns, churn prediction has to be extremely accurate, to ensure that the incentives reach only the customers who are most likely to change their service providers.

Building a churn prediction model will ease the customer retention process, and in this way the mobile telecommunications companies will success in this constantly increasing competitive market. The churn prediction modeling process is strongly dependent on the data mining process and techniques due to an increased performance generated by machine learning algorithms compared to the statistical techniques for nonparametric data [4].

Data mining is the practice of digging data to find trends and patterns, and can provide you with answers to questions that you should have asked. Data mining methods lie at the intersection of artificial intelligence, machine learning, statistics, and database systems [5]. Data mining techniques can help building prediction models in order to discover future trends and behaviors, allowing organizations to make smart decisions based on knowledge from data.

The methodology used is called modeling. Modeling is the process of creating a mining model, an implementation of specific data mining algorithms. These algorithms are mathematical functions that perform specific types of analysis on the associated dataset. Based on the business problems that need assistance, these data mining algorithms can be broadly classified into several categories [6]. The category which is of interest for us is called classification analysis. Classification is the process by which a model is built to categorize pre-classified instances (training examples) into classes. This classifier (classification model) is then used to classify future instances.

This paper is organized as follows. In Section 2, we describe the k-nearest neighbor, logistic regression, and bayesian networks algorithms that are later used to build the predictive models. In Section 3, we present the methodology and test the models on a data set containing call details records. Finally, Section 4 concludes this research work.

II. DATA MINING ALGORITHMS

In this section, we present the algorithms that we will use to build the predictive models in order to solve the subscribers churning problem.

Manuscript received April 26, 2013; revised September 25, 2013

A. K-NEAREST NEIGHBORS

The k-Nearest Neighbor (kNN) algorithm is proposed by Fix and Hodges (1951) [7]. The kNN algorithm is a non-parametric lazy learning algorithm and represents a method for classifying cases based on their similarity to other cases. The similar cases, called neighbors, are near each other, while dissimilar ones are distant. For a new case to be classified, the distance between it and each of the cases in the model is computed and is placed into the class most common amongst its *k* nearest neighbors.

In order to prevent differences in measurement scale we need to preprocess the features. We code the continuous features by using adjusted normalization:

$$x_{pn} = \frac{2(x_{pn}^0 - \min(x_p^0))}{\max(x_p^0) - \min(x_p^0)} - 1 \quad (1)$$

where x_{pn} represents the normalized value of feature p for case n, x_{pn}^0 the original value for case n, $\min(x_p^0)$ and $\max(x_p^0)$ the minimum, respectively, the maximum value for all training cases.

We code the categorical features by using one-of-*c* coding – for a feature with *c* categories we have *c* vectors, with first category coded as (1,0,…,0), the second as (0,1,…,0), and so on.

To train the nearest neighbor model, meaning to compute the distances between cases based upon their values, we choose the Euclidian metric:

$$d_{ih} = \sqrt{\sum_{p=1}^{P} w_{(p)}(x_{(p)i} - x_{(p)h})^2} \quad (2)$$

where $w_{(p)}$ is the feature weight which is equal to the normalized feature importance, as shown in equation (3):

$$w_{(p)} = FI_{(p)} / \sum_{p=1}^{P} FI_{(p)} \quad (3)$$

To compute the feature importance, $FI_{(p)}$ we use equation (4) and evaluate the error rate $e_{(p)}$ (5):

$$FI_{(p)} = e_{(p)} + \frac{1}{p} \quad (4)$$

$$e_{(p)} = \left(1 - \sum_{j=1}^{J} \hat{N}_j / N\right) \times 100\% \quad (5)$$

where N represents the number of cases and N_j the number of cases that belongs to class j and are correctly classified as j.

We classify a case by a majority vote of its *k* nearest neighbors (previously found) among the training cases. If there is a tie on the predicted probability, we choose the category with the largest number of cases in the training data set. If we still have a tie on the largest number of cases, we choose the category with the smallest data value.

B. LOGISTIC REGRESSION

The logistic regression algorithm is a well-established statistical technique used to classify objects based on the values of input fields. Compared to linear regression, this algorithm has a categorical target variable. The equations involved in the logistic regression algorithm relate the values of the predictor variables to the probabilities associated with each of the output category. After the model is generated, probabilities for new data can be estimated. A new record is predicted as belonging to the category with the highest computed membership probability.

The linear logistic model assumes a dichotomous target variable Y, with probability π. The likelihood function l has the equation (6):

$$l = \prod_{i=1}^{n} \pi_i^{w_i y_i} (1 - \pi_i)^{w_i (1 - y_i)} \quad (6)$$

where n represents the number of cases, y_i the observations, π_i the probabilities for y_i, w_i the weight for the i^{th} case. To compute the probability for the i^{th} observation we use equation (7):

$$\pi_i = \frac{\exp(\eta_i)}{1 + \exp(\eta_i)} \quad (7)$$

where $\eta_i = \mathbf{X}_i' \beta$, β is a $p \times 1$ vector with element β_j (the coefficient for the j^{th} parameter) and $\mathbf{X}$ a $n \times p$ matrix with x_{ij} element (the observed value of the i^{th} case of the j^{th} parameter).

Therefore, to compute the log-likelihood function L, we use equation (8):

$$L = \ln(l) = \sum_{i=1}^{n} (w_i y_i \ln(\pi_i) + w_i (1 - y_i) \ln(1 - \pi_i)) \quad (8)$$

To obtain the Maximum Likelihood Estimates (MLE) we use the Newton-Raphson method as described in [8]. The convergence is based on the parameter estimates difference between the iterations, on the log-likelihood function difference between successive iterations, and on the number of iterations.

After we obtain the maximum likelihood estimates $\hat{\beta}$, we estimate the asymptotic covariance matrix by I^{-1}, where:

$$I = -\left[E\left(\frac{\partial^2 L}{\partial \beta_i \partial \beta_j}\right)\right] = \mathbf{X}'\mathbf{W}\hat{\mathbf{V}}\mathbf{X} \quad (9)$$

$$\hat{\mathbf{V}} = Diag\{\hat{\pi}_i (1 - \hat{\pi}_i)\} \quad (10)$$

$$\mathbf{W} = Diag\{w_i\} \quad (11)$$

$$\hat{\pi}_i = \frac{\exp(\hat{\eta}_i)}{1 + \exp(\hat{\eta}_i)} \quad (12)$$

$$\hat{\eta}_i = \mathbf{X}_i' \hat{\beta} \quad (13)$$

If $\hat{\pi}_i > 0.5$, the predicted of y_i is 1, otherwise is 0.

C. BAYESIAN NETWORKS

The Bayesian network model is proposed by Judea Pearl (1988) [9]. A Bayesian network represents the joint probability distribution for a given a set of categorical random variables (*V*). A Bayesian network has as constituent parts the directed acyclic graph *G = (V, E)* (where *E* is a set of directed edges) and a conditional probability table (CPT) for each node given the values of its parent nodes. Given the value of its parents, each node is assumed to be independent of all the nodes that are not its descendants. The joint probability distribution for variables *V* can then be computed as a product of conditional probabilities for all nodes, given the values of each node's parents.

Given the set of variables *V* and a corresponding sample dataset, we are presented with the task of fitting an appropriate Bayesian network model. The task of determining the appropriate edges in the graph *G* is called *structure learning*, while the task of estimating the conditional probability tables given parents for each node is called *parameter learning*.

When building the Bayesian network model we choose the Tree Augmented Naïve Bayes (TAN) method [10]. Compared with general Bayesian network models this method has a better classification accuracy and performance. This model works only with discrete variables, and therefore we discretize the continuous variables into 5 equal-width bins.

This method classifies a case $d_j = (x_1^j, x_2^j, ..., x_n^j)$ by calculating the probability of it belonging to the target variable Y_i, as shown in equation (14):

$$\Pr(Y_i|X_k = x_k^j) \propto \Pr(Y_i)\prod_{k=1}^{n}\Pr(X_k = x_k^j|\pi_k^j, Y_i) \quad (14)$$

where Y is the categorical target variable, X_i the i^{th} predictor, π_k the parent set of X_k besides Y, and $\Pr(X_k|\pi_k,Y)$ represents the CPT associated with each node X_k.

The TAN classifier [10] that we use is defined by the following two conditions: first, each predictor has the target as a parent, and second, the predictors may have one other predictor as a parent. The TAN learning procedure takes as input the training data *D*, the categorical predictor vector $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ and *Y*. First, it learns a tree network structure over $\mathbf{X}$ by using the *TAN structure learning* algorithm.

During the structure learning phase, we use the maximum weighted spanning tree (MWST) method to construct a tree Bayesian network from data [11]. This method associates a weight to each edge corresponding to the mutual information between two nodes (X_i, X_j) shown in equation (15). When the weight matrix is created, the MWST algorithm [12] gives an undirected tree that can be oriented with the choice of a root.

$$I(X_i, X_j) = \sum_{x_i, x_j}\Pr(x_i, x_j)\log\left(\frac{\Pr(x_i, x_j)}{\Pr(x_i)\Pr(x_j)}\right) \quad (15)$$

Replacing the mutual information with the conditional mutual information between two nodes given the target [10], we obtain equation (16):

$$I(X_i, X_j|Y) = \sum_{x_i, x_j, y_k}\Pr(x_i, x_j, y_k)\log\left(\frac{\Pr(x_i, x_j|y_k)}{\Pr(x_i|y_k)\Pr(x_j|y_k)}\right) \quad (16)$$

The network is constructed by computing the conditional mutual information between each pair of variables. Then, by applying Prim's algorithm [12] we construct an MSWT with a weight of an edge connecting X_i to X_j by $I(X_i,X_j|Y)$. The resulted undirected tree is transformed to a directed one by choosing X_1 as a root node and setting the direction of all edges to be outward from it. After the network structure has been learned the target variable Y is added as a parent of every X_i, where $1 \le i \le n$.

The last part of the TAN learning procedure is the *TAN parameter learning*. During parameter learning phase, we use the maximum likelihood estimation method. The closed form solution for the parameters $\theta_{Y_i}(1 \le i \le |Y|)$ and $\theta_{ijk}(1 \le i \le n, 1 \le j \le q_i, 1 \le k \le r_i)$ that maximize the log-likelihood score is:

$$\begin{aligned}\hat{\theta}_{Y_i} &= N_{Y_i}/N \\ \hat{\theta}_{ijk} &= N_{ijk}/N_{ij}\end{aligned} \quad (17)$$

where $r_i = |X_i|$ is the cardinality of X_i, $q_i = r\pi_i \times |Y|$ is the cardinality of the parent set π_i of X_i, N_{ij} is the number of records in *D* for which (π_i, Y) takes the j^{th} value, N_{ijk} is the number of records in *D* for which (π_i, Y) takes the j^{th} value and for which X_i takes the k^{th} value, N_{Y_i} the number of cases with $Y = Y_i$ in the training data, and *K* the number of parameters given by equation (18):

$$K = \sum_{i=1}^{n}(r_i - 1)\cdot q_i + |Y| - 1 \quad (18)$$

For the *TAN Posterior Estimation*, we assume that Dirichlet prior distributions are specified for the sets of parameters [13]. The Dirichlet posterior distributions have the following sets of parameters (19):

$$\begin{aligned}\hat{\theta}^P_{Y_i} &= (N_{Y_i} + N^0_{Y_i})/(N + N^0) \\ \hat{\theta}^P_{ijk} &= (N_{ijk} + N^0_{ijk})/(N_{ij} + N^0_{ij})\end{aligned} \quad (19)$$

where $N^0_{Y_i}$ and N^0_{ijk} denote corresponding Dirichlet distribution parameters such that:

$$\begin{aligned}N^0 &= \sum_i N^0_{Y_i} \\ N^0_{ij} &= \sum_k N^0_{ijk}\end{aligned} \quad (20)$$

To overcome problems caused by zero or very small cell counts, parameters can be estimated as posterior parameters $\hat{\theta}^P_{Y_i}(1 \le i \le |Y|)$ and $\hat{\theta}^P_{ijk}(1 \le i \le n, 1 \le j \le q_i, 1 \le k \le r_i)$ using

uninformative Dirichlet priors:

$$N_{Y_i}^0 = 2/|Y| \qquad N_{ijk}^0 = 2/r_i q_i \tag{21}$$

III. METHODOLOGY

In this section we will present a methodology which the data mining approach proposed in this paper is based on. The first part of this section implies understanding and preparing the dataset and the second one building and evaluating the models.

A. DATA PREPARATION

The dataset is from University of California, Department of Information and Computer Science, Irvine, CA [14]. This dataset contains historical records of customer churn, how they turned out in hindsight, i.e., their previous behavior – if it turned out that they are churners or not.

This dataset has a total number of 3333 subscribers with 21 variables each. For each of these subscribers we can find information about their corresponding inbound/outbound calls count, inbound/outbound SMS count, and voice mail.

For the data preparation and model building part we decided to use IBM SPSS (Statistical Product and Service Solutions), a statistical and data mining software package used to build predictive models [15].

By exploring the data we discovered that this is a complete dataset, meaning that for each subscriber there is no attribute missing. We also discovered that between some variables there is a perfect correlation with the R-squared statistic precisely 1. We have four *charge* variables which are linear functions of the *minutes* variables, so we decided to arbitrarily eliminate all four of them to avoid incoherent results. The *area code* variable contains only three different values (from CA only) for all the records, while the *state* variable contain all 51 states, so we decided not include these two variables as well, as it can be bad data. The *phone* variable has been eliminated because it does not contain relevant data that can be used for prediction; it is useful only for identification purposes. We have therefore reduced the number of predictors from 20 to 13. The target variable is *Churn*, which has two values, one of them for each subscriber: *Yes* or *No*, telling if a subscriber is a churner or not. Table 1 lists all attributes together with their desired corresponding type.

Because we intend to build a classification model based on learning algorithms we must train them first and then test them, and therefore we must partition the dataset in two: a training and a testing set [16]. After multiple partitioning attempts we concluded that the best performance is obtained if we randomly partition the training set to be approximately 80% of the original data set, consisting of 2651 subscribers, and the testing set to be approximately 20%, consisting of 682 subscribers, as shown in Figure 1.

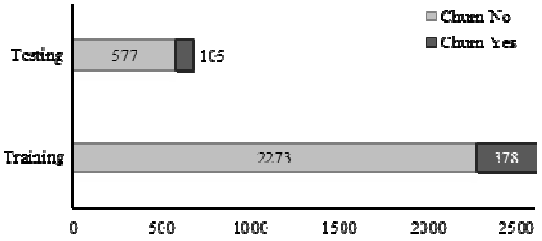

Figure 1. Samples and churn distributions.

The churn distribution within both sets is as follows: in the testing set we have 105 churners (15%) and 577 non-churners (85%), and in the training set we have 378 churners (14%) and 2273 non-churners (86%). Because the algorithms require to have a nearly equal distribution of "*Yes-es*" and "*No-s*" to train properly, we need to clone the "*Yes-es*" until they are approximately equal to the "*No-s*" by balancing by boosting [17] the training set. Therefore, we will obtain a distribution of 2273 records for churners and 2273 records for non-churners, a total of 4546 records within the training set, as shown in Figure 2.

Used Variables

Variable Name	*Type*	*Values*
Account Length	Continuous	1.0 – 243.0
International Plan	Categorical	Yes/No
Voice Mail Plan	Categorical	Yes/No
# Voice Mail Messages	Continuous	0 – 51.0
# Day Minutes	Continuous	0 – 350.8
# Day Calls	Continuous	0 – 165.0
# Evening Minutes	Continuous	0 – 363.7
# Evening Calls	Continuous	0 – 170.0
# Night Minutes	Continuous	23.2 – 395.0
# Night Calls	Continuous	33.0 – 175.0
# International Minutes	Continuous	0 – 20.0
# International Calls	Continuous	0 – 20.0
Customer Service Calls	Continuous	0 – 9.0
Churn?	Categorical	Yes/No

Omitted Variables

Variable Name	*Type*	*Values*
State	Categorical	AK, AL, …
Area Code	Categorical	N/A
Phone	Categorical	N/A
# Day Charge	Continuous	0 – 59.64
# Evening Charge	Continuous	0 – 30.91
# Night Charge	Continuous	1.04 – 17.77
# International Charge	Continuous	0 – 5.4

Table 1. Data set variables.

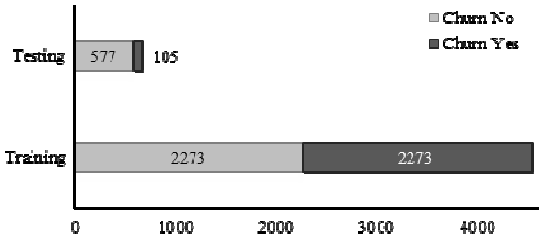

Figure 2. Samples and churn balanced distributions.

B. MODEL BUILDING

This phase involves choosing the modeling algorithms and modeling architecture, and finally building and evaluating the model.

As we mentioned in the previous section, we will use k-nearest neighbor, logistic regression, and bayesian networks algorithms to implement the model. For each of these algorithms we tested multiple configurations. In kNN algorithm's case, the highest predicting performance is achieved if we use the previously described algorithm with k=14 nearest neighbors and the Euclidian metric for distance computation. The features are weighted by importance when computing distances.

For logistic regression algorithm, the algorithm described in section 2, with the following convergence criteria scored the highest performance: for parameter convergence we use the value 1.0E-8, for log-likelihood convergence we use the value 1.0E-3, and the maximum number of iterations is set to 20.

We can visualize the performance of the models on the testing set, by looking at the confusion matrices in Table 2. The cells on the diagonal of the cross-classification of cases are correct predictions, whilst those off the diagonal are incorrect predictions.

		Predicted	
Observed	*No*	*Yes*	*Overall %*
No	496	81	85.96
Yes	17	88	*83.81*
Overall %	75.22	24.78	85.63

Table 2.a. Confusion matrix for kNN model.

		Predicted	
Observed	*No*	*Yes*	*Overall %*
No	449	128	77.82
Yes	20	85	*80.95*
Overall %	68.77	31.23	78.30

Table 2.b. Confusion matrix for logistic regression model.

		Predicted	
Observed	*No*	*Yes*	*Overall %*
No	496	81	85.96
Yes	21	84	*80.00*
Overall %	75.81	24.19	85.04

Table 2.c. Confusion matrix for bayesian networks model.

Table 2. Confusion matrices.

This suggest that, overall, all our three models will correctly classify approximately 8 out of 10 subscribers as churners (see Table 2, where *Predicted – Overall %* intersects *Observed – Yes*). We can see that, in our case, all three algorithms have almost the same accuracy.

We can compare the performance of our models by using the gain measure. The gain chart shows on the vertical axis the percentage of positive responses, while on the horizontal axis the percentage of customer contacted. Gains are defined as the proportion of respondents that occurs in each percentile, relative to all responders in the whole sample. The cumulative gain chart shows how better the prediction rate produced by our models is, compared to the random expectation (diagonal line in Figure 3).

Figure 3 depicts the curves corresponding to the models that use k-nearest neighbor (KNN), logistic regression (LOGR), and bayesian networks (BN) algorithms. One can find that in the 20th percentile, the models that use k-nearest neighbor and bayesian networks algorithms perform better than the one that uses logistic regression algorithm, approximately 80% for KNN and BN, compared to approximately 60% for LOGR.

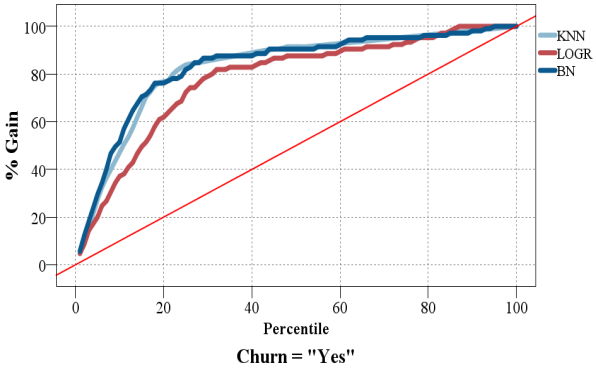

Figure 3. Cumulative gain chart for kNN, LogR, and BN.

Another way of interpreting the results are the lift charts. The lift chart sorts the predicted pseudo-probabilities [18] in descending order and display the corresponding curve. There are two types of lift charts: cumulative and incremental. The cumulative lift chart shows how better the prediction rate produced by our models is, compared to the random expectation.

Figure 4 depicts the curves corresponding to KNN, LOGR, and BN models, and by reading the graph on the horizontal axis, we can see that for the 19th percentile, both KNN and BN models have approximately a 4 lift index on the vertical axis.

The incremental lift chart displayed in Figure 5, shows the lift in each percentile without any accumulation. The lift line corresponding to KNN model descends below the random line (red straight line) at about 19th percentile, meaning that compared to random expectation this model achieve its best performance in the first 19% of the records. The BN model achieves its best performance in the first 16% of the records, while LOGR model in the first 11% of the records, as we can see in Figure 5.

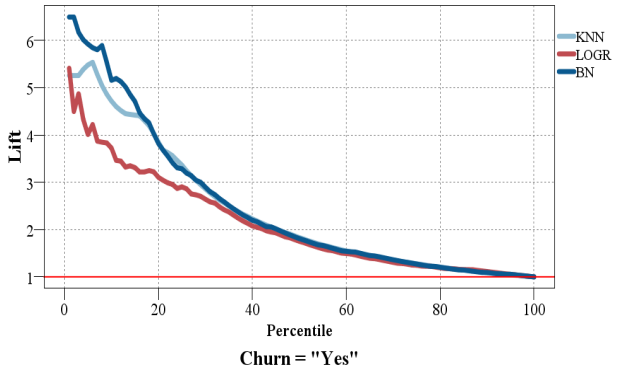

Figure 4. Cumulative lift chart for kNN, LogR, and BN.

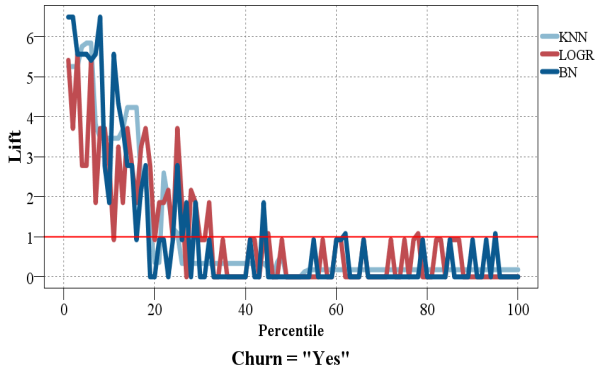

Figure 5. Incremental lift chart for kNN, LogR, and BN.

If, for instance, a mobile telecommunications company decides to send offers containing incentives to stay with them, they can easily select, if for example they use the gain chart, the top 20% subscribers (20th percentile), and expect that 80% of the contacted subscribers to be churners (if they use kNN model or bayesian networks model). If they use the lift chart, they can select the top 16-19% subscribers in this sorted scored list, and expect to contact more than four times (lift index 4) the number of subscribers that are categorized as churners than normal (again, if they use kNN model or bayesian networks model).

IV. CONCLUSIONS

In this paper we built three predictive models for subscribers churn in mobile telecommunications companies, using k-nearest neighbor, logistic regression, and bayesian networks algorithms. By evaluating the results, from the technical point of view, we observe that for predicting both churners and non-churners, the model that uses the k-nearest neighbor algorithm performs best, having an overall accuracy of 85.63%.

From the practical point of view, all three models have a very good performance (around 80%) in predicting churners in a mobile telecommunications company. Decision making employees can build different marketing approaches to retain churners based on the predictors that have higher importance in scoring the model performance. All these churn prediction models can be used in other customer response models as well, such as cross-selling, up-selling, or customer acquisition.

REFERENCES

[1] M. Shaw, C. Subramaniam, G.W. Tan, and M.E. Welge, "Knowledge management and data mining for marketing", *Decision Support Systems*, Vol. 31, no. 1, pp. 127-137, 2001.

[2] C.P. Wei and I.T. Chiu, "Turning telecommunications call details to churn prediction: a data mining approach", *Expert Systems with Applications*, Vol. 23, pp. 103-112, 2002.

[3] J.H. Ahn, S.P. Han, and Y.S. Lee, "Customer churn analysis: Churn determinants and mediation effects of partial defection in the Korean mobile telecommunications service industry", *Telecommunications Policy*, Vol. 30, Issues 10–11, pp. 552-568, 2006.

[4] V. García, A.I. Marqués, and J.S. Sánchez, "Non-parametric Statistical Analysis of Machine Learning Methods for Credit Scoring", *Advances in Intelligent Systems and Computing*, Volume 171, pp. 263-272, 2012.

[5] S. Chakrabarti, M. Ester, U. Fayyad, J. Gehrke, J. Han, S. Morishita, G. Piatetsky-Shapiro, and W. Wang, "Data Mining Curriculum: A Proposal", Version 1.0, 2006.

[6] F. Gorunescu, *Data Mining Concepts, Models and Techniques*, Springer-Verlag Berlin Heidelberg, 2011.

[7] E. Fix and J.L. Hodges Jr., "Discriminatory analysis, nonparametric discrimination", *USAF School of Aviation Medicine*, Randolph Field, 1951.

[8] S.A. Czepiel, "Maximum likelihood estimation of logistic regression models: theory and implementation", Available: http://czep.net/stat/mlelr.pdf [Accessed: February 9, 2013].

[9] J. Pearl, *Bayesian Networks*, Computer Science Department, University of California, Los Angeles, 1988.

[10] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers", *Machine Learning*, Vol. 29, pp. 131–163, 1997.

[11] C.K. Chow and C.N. Liu, "Approximating discrete probability distributions with dependence trees", *IEEE Transactions on Information Theory*, Vol. 14, pp. 462–467, 1968.

[12] R.C. Prim, "Shortest connection networks and some generalizations", *Bell System Technical Journal*, Vol. 36, pp. 1389–1401, 1957.

[13] D. Heckerman, "A Tutorial on Learning with Bayesian Networks", *Learning in Graphical Models*, MIT Press, pp. 301–354, 1999.

[14] C.L. Blake and C.J. Merz, "Churn Data Set", *UCI Repository of Machine Learning Databases*, http://www.ics.uci.edu/~mlearn/MLRepository.html, University of California, Department of Information and Computer Science, Irvine, CA, 1998.

[15] IBM SPSS Modeler, IBM Corporation, http://www-01.ibm.com/software/analytics/spss/

[16] S. Sumathi and S. N. Sivanandam, *Introduction to Data Mining and its Applications*, Studies in Computational Intelligence, Vol. 29, Springer, 2006.

[17] R. Nisbet, J. Elder IV, and G. Miner, *Handbook of Statistical Analysis and Data Mining Applications*, Academic Press, 2009.

[18] L. Nedovic, B. Mihailovic, and N. M. Ralevic, "Some properties of pseudo-measures and pseudo-probability", *Intelligent Systems and Informatics*, pp.155-159, 2007.