International Journal of Advanced Science and Technology
Vol.120 (2018), pp.13-28
http://dx.doi.org/10.14257/ijast.2018.120.02

Cost-Sensitive Genetic Programming for Churn Prediction and Identification of the Influencing Factors in Telecommunication Market

Nailah Al-Madi[1*], Hossam Faris[2] and Ruba Abukhurma[2]

[1]*The King Hussein Faculty of Computing Sciences, Princess Sumaya University for Technology, Amman, Jordan*
[2]*King Abdullah II School for Information Technology, the University of Jordan, Amman, Jordan*
n.madi@psut.edu.jo, hossam.faris@ju.edu.jo, ruba_abukhurma@yahoo.com

Abstract

Churn prediction is crucial for companies in order to build an efficient customer retention plans and apply successful marketing strategies. However, from data mining point of view, churn prediction is considered as a complex and complicated task due to the highly imbalanced distribution of the class labels where the ratio of the churning customers are much smaller than the ratio of the loyal ones. Genetic programming has proven its efficiency in prediction problems, since it solves them as an optimized classification problem with the ability of identifying the relevant features. In this paper, we propose the application of a genetic programming with cost sensitive learning (GP-CSL) that solves the churn prediction task as a classification problem with a penalty cost for prediction errors. Comprehensive experiments on real telecommunication dataset are applied using the proposed GP-CSL technique to evaluate its effectiveness in predicting the customers vulnerable to churn. The obtained results indicate promising churn detection rates compared with other well-known classifiers. Moreover, a formal analysis of the most relevant features in the dataset is performed using different penalty cost matrices and other classical fitness functions. The analysis results show that 4 to 5 features out of 10 features are the most relevant ones. In terms of business, this could help decision makers to determine the most influential factors on customers churn, and consequently help them in planning effective retention campaigns.

Keywords: *Churn Prediction, Genetic Programming, Cost Sensitive Learning, Telecommunication*

1. Introduction

A key issue in any customer relationship management (CRM) is the effectiveness of the adopted customer retention strategy. In this context, accurate and early prediction of which customers are going to churn is very important for planning efficient marketing campaigns and consequently successful retention strategies [1]. The importance of customer retention is that in most cases the cost of acquiring new customers is much greater than retaining existing customers. Successful companies realize this importance which can be seen nowadays in different sectors such as telecommunication markets, banking and insurance companies [2]. Therefore, predicting customer churn is considered as a very essential and vital process in business marketing.

Received (May 15, 2018), Review Result (July 23, 2018), Accepted (August 5, 2018)
* Corresponding Author

ISSN: 2005-4238 IJAST

Conventionally, churn prediction is tackled as a binary classification problem, where the goal is to predict whether the customer will leave the company (*i.e.*, churn) or not (*i.e.*, loyal) [3]. In the literature, different classical machine learning algorithms were applied and investigated for this task such as Artificial Neural Networks (ANNs) [4], Support Vector Machines (SVMs) [5], Decision Trees (DT) [6], and Logistic Regression (LR) [6]. Hybridization of more than one algorithm were also proposed churn prediction in an attempt to outperform the single algorithm approach [7].

From a machine learning perspective, customer churn prediction is considered a very challenging and complex problem. This is due to the highly imbalanced distribution of the class labels of the data, where in telecommunication market the percentage of churners ranges from 5% to 15% in most cases [8]. This problem makes the application of standard machine learning algorithms inefficient, because they tend to give more attention to majority class and disregard the rare class, which represents the churn customers [9].

In this work we propose the application of a genetic programming approach with cost sensitive learning (GP-CSL) for customers churn prediction in the telecommunication market. Genetic programming is considered one of the popular optimization techniques that successfully applied for solving different classification problems [10, 11]. In this paper, the cost of the churner's class, which represents the rare class is controlled and increased to guide the generated GP models to give more attention toward churner's class. The used costs of churners can be in a way that is more flexible with the costs of the retention campaigns. Another motivation for using GP is the automatic feature selection mechanism that embedded in its design. GP can give an insight on the most influencing features in the classification. Identifying such features can significantly help decision makers in planning effective customer retention campaigns.

This paper aims to achieve high churn prediction rates, and utilize the embedded feature selection mechanism in GP to identify the most influencing features of the churn prediction problem. The proposed GP-CSL model is verified and tested using a real data set, which is collected from a major Jordanian telecommunication company. In addition, the effect of different cost values are experimented on the churn prediction accuracy. Furthermore, a formal analysis of the most relevant features in the dataset is performed using different penalty cost matrices and other classical fitness functions. In addition, the developed model is evaluated and compared with different machine learning models that commonly applied in the literature for churn prediction problem.

This paper is organized as follows: Section 2 discusses the related work done in classification for imbalanced data, especially for customer churn prediction. Section 3 describes the genetic programming and how it is used to solve classification problems. In Section 4 the proposed GP-CSL is explained. Section 5 describes the model evaluation and the experimental results. Finally, the findings and remarks of this work, and future works are concluded in Section 6.

2. Related Work

In machine learning, the approaches that handle the problem of the imbalanced class distribution can fall into four main approaches. The first one is an internal approach, where the learning algorithms are modified internally to overcome this problem. An example of this approach for churn prediction can be found in [5], where the authors introduced an improved one-class SVM method for churn prediction with different kernel functions The authors proved that using SVM for churn prediction is an efficient, but it needs a lot of time to select proper kernel parameters and input features.

The second one is an external approach (also known as data level approach), where the training data is preprocessed to decrease the effect of the difference between class ratios by using the oversampling and under-sampling techniques [12].

The third approach is the cost sensitive learning, which assigns misclassification penalty to each class. The authors in [13] proposed a financial based measure for evaluating the effectiveness of a churn campaign taking by considering offers information and their financial cost and probability acceptance that depending on the customer profile. Moreover, they used a real-world churn dataset to compare different cost-sensitive classification algorithms (decision tree, random forests, and logistic regression). The results show that cost-sensitive approach can rise the cost savings.

Another example of using cost-sensitive approach can be found in [14]. The authors introduced the notion of customer lifetime value (CLV), which could be defined as the discounted value of future marginal earnings, based on the customer's activity, where a churner is defined as someone whose CLV is decreasing. The results showed that the cost-sensitive approaches achieved very good results comparing to the other five churn prediction methods namely; Logistic Regression, Decision Trees, Neural Networks, AdaCost, and cost sensitive Decision tree in terms of the defined profit measure besides achieving a good overall classification.

The fourth approach is based on using ensembles of classifiers that combine the prediction of a group of classifiers [15]. The authors in [16] applied four classification techniques: Decision Tree, Artificial Neural Networks, K-Nearest Neighbors, and Support Vector Machine to compare their performances using data from Iranian mobile company. After that, the authors proposed a hybrid methodology, which made considerable improvements in terms of prediction accuracy, recall, and precision.

Another example of using the ensemble approach was presented in [17], where the authors proposed a one-step classifier ensemble model for imbalanced data. This ensemble method is adaptive such that it can select one of two kinds of dynamic ensemble approaches (dynamic classifier selection and dynamic ensemble selection). In the same research line, the authors in [15] proposed a negative correlation learning (NCL) based ensemble of neural networks for predicting churn in a telecommunication company. The results proved that the used approach can achieve better churn detection than other popular machine learning algorithms.

As noted in most of the related work, cost-sensitive approaches have been investigated and successfully applied on churn prediction problem. The significant merits of the cost-sensitive approach motivated our attempts to apply it to the GP and investigate its effectiveness on customer churn prediction problem. In this paper, GP is applied on a real telecommunication churn prediction problem. The GP capabilities are used to locate the best classifier that suits the churn problem. Furthermore, the training process is accomplished with the use of cost sensitive approach by learning that misclassifying a churn customer has a high penalty cost. Moreover, GP is used to identify the most relevant features that may affect the customer to churn

3. Genetic Programming

Genetic Programming (GP) [18] is one of the evolutionary computation approaches that is used as a powerful tool for optimization and problem solving, and has been applied to a wide range of applications [19]. GP is an effective technique, since it automatically solves problems without requiring the user to know the structure of the solution in advance. Thus, it can be used, whenever both the solution and its structure are unknown to the user. Moreover, GP is distinguished from other evolutionary algorithms by the use of a variable size tree representation rather than a linear fixed length representation. GP solves the problems by generating programs, and these programs consist of mathematical functions (+, -, *, / ... *etc.*), logical functions (ifElse, largerThan, equals ... *etc.*), and terminals, which can be variables or constants. GP programs are evaluated by fitness function, which represents a function that asses the goodness/fitness of the solution that provided by each program. As an optimization

process GP's main objective is to maximize or minimize the fitness value based on the target problem. The GP process has four main steps which can be described as follows:

1. First, a random population of programs (solutions) is generated.
2. Then, a fitness ranking is calculated for each program using the fitness function defined based on the problem to be solved.
3. After that, a selection process is applied to choose two programs (parents) in order to generate new programs (children). Selection applies the natural rule of survival of the fittest, which ensures the best solution quality in the optimization problems.
4. Then, the new programs (children) are generated by applying natural operations of crossover and mutation, which ensure the diversity among solutions.

All these steps are repeated until a new population with the same size as the old population is produced, as shown in the flowchart represented in Figure 1. This process is applied over and over until a stopping criterion is satisfied, such as reaching a predefined number of generations, or finding a solution with the predefined fitness. Many important parameters must be defined before starting the GP run, for example; the fitness function to evaluate the program quality, population size, number of generation, functions and constants to be used in generated programs (solutions), and crossover and mutation probabilities. The details of the previous steps are discusses in the following subsections.

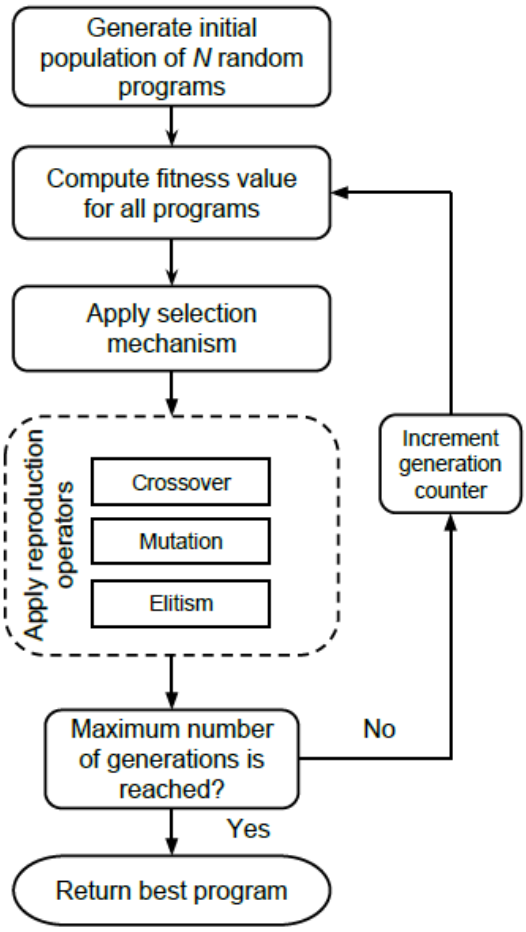

Figure 1. GP Flowchart

3.1. GP Programs and Classification Problems

GP programs are similar to genetic algorithm chromosomes, however, these programs has variable size and formatted of different functions and values, whereas GA chromosomes has fixed size and their genes are well formatted to contain known values (binary, integers...*etc.*).

An example of a tree representation for a GP program is shown in Figure 2, where the program consists of three functions (plus, sin, division) and two variables (x and y) and one constant value (4), this program can be written in LISP language as: (+(sin x) (/ y 4)).

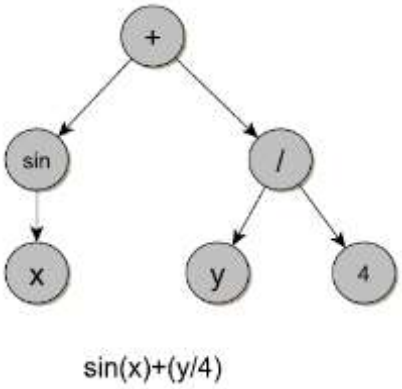

Figure 2. An Example of GP Program

GP is used to solve a classification problem such that each program is a suggested classifier. To evaluate the classifier's quality, we can use any classification quality measure as a fitness function such as classification accuracy, which represents the percentage of correctly classified instances. Therefore, to calculate the fitness value of the program shown in Figure 2, (assuming that x and y are values of data instance), the program is executed and get the resulted number, then this number is converted to a class label and compare it with the original class label. Converting the result to a class label can be done using different ways, such as calculating the modulus of the results over the number of classes (if the problem has two classes A, and B, then the result would be 0 or 1, where 0 can be class A, 1 can be class B).

The process of executing the program is done for each instance of the data and then the accuracy of the program is computed as a fitness value. The GP process continues until finding the best classifier with the highest accuracy. Different fitness functions can be used such as Mean Squared Error (MSE) and Pearson (R^2), where GP calculates one of them for each program instead of accuracy and searches for the program with the best fitness value (minimum error).

3.2. Natural Operations

Three natural operators are applied in GP (selection, crossover, and mutation). Selection process is the process of selecting two programs as parents to generate new programs as children. There are many selection methods in the literature such as: Roulette wheel selection, Tournament selection, and Rank selection [20]. Exploration and exploitation of the search space to find a better program is done using crossover and mutation operators. Crossover means switching one node of a program with another node from another program of the parents. With a tree-based representation, replacing a node means that the whole branch is being replaced. However, Mutation means changing a single program by either replacing a function with another function or replacing a terminal with another terminal. Crossover and mutation are applied based on a predefined probability, such that based on random value either crossover is applied or not, same applies for mutation. Usually crossover probability is higher than mutation probability.

4. Proposed GP-CSL Based Framework

GP searches for the best fitness value of the program, whether a minimum or maximum value. The fitness function evaluates the quality of the proposed solution and is the core point that controls the GP process. Therefore, to find the optimal solution for a problem we need a good fitness function that fits the problem. Traditionally, researchers solve classification problems using GP with accuracy as a fitness function (searching for the maximum value). Classification accuracy means the percentage of correctly classified instances. However, when the data of the problem to be solved is imbalanced, accuracy is not a good option to be used as it may suffer from over-fitting problem (give more attention to majority class). Therefore, many techniques were used to overcome over-fitting, and at the same time build a good classifier for imbalanced data, one of these techniques is the cost sensitive learning [21].

Cost sensitive learning aims to give a higher cost for predicting a specific class, such as any misclassification of a class will penalize (cost more) the classifier. Cost matrix is a matrix that describes the cost for misclassification in a particular scenario. It is similar to confusion matrix, however, in a binary classification case, its values are the cost of classifying an instance from class 1 into 1, or into 0, and an instance from class 0 into 1 or 0. Where it gives higher cost to misclassifying rather than correctly classify the instance. Table 1, shows an example of a cost matrix, where classifying a nonChurner instance as a churner, will cost the classifier a penalty equal to $Cost_c$. A penalty cost equals to $Cost_B$ can be defined also for classifying a churner instance as nonChurner. Defining high cost values gives more importance to that class. Therefore, it helps in building a classifier for imbalanced data and can help in training the classifier to detect the class with the less number of instances and avoid the over-fitting problem. In this paper we adjust the GP to use the cost sensitive learning approach as its fitness, rather than the traditional accuracy. Therefore, the fitness function can be calculated as shown in equation 1:

$$\text{CSL-Fitness} = Cost_c * C + Cost_B * B \quad (1)$$

Where $Cost_c$ is the penalty cost of classifying a nonChurner as a churner, C is the number of instances that were classified as a churners and they are not. $Cost_B$ is the penalty cost of classifying a churner as a nonChurner, B is the number of instances that are churners but were classified as nonChurners. When GP-CSL uses this equation as a fitness function then it searches for the minimum value retrieved.

Table 1. Cost Matrix

	Actual	
	nonChurners	Churners
Predicted nonChurners	0	$Cost_B$
Predicted churners	$Cost_c$	0

4.1. Identification of the Most Relevant Features

One of the most important features of GP is the interpretability of the generated models. In addition, GP can give an insight on the most relevant variables using an embedded feature selection mechanism. Over the course of the evolutionary cycle of GP, features that have more impact on the accuracy of the classification models will survive and keep appearing in the models, while the less relevant ones will decay in the models until they disappear. To identify the most relevant features in our developed GP based churn prediction model, we refer to the “relative variable relevance” measurement [22]. The relative variable relevance for each variable is calculated based on the number of variable

references in all models generated over the course of GP iterations. The relative variable relevance of a given feature can be described in the following steps:

- Suppose S is a set of input variables $\{x_1, x_2 \dots x_n\}$ where $x_i \in S$.
- The frequency of x_i with respect to a given model M generated during the GP evolutionary cycle can be defined as RefCount(x_i , M) which is the number of references to x_i in M.
- Thus, the frequency of x_i with respect to a population P can be defined as:

$$\text{freq}(x_i, P) = \sum_{M \epsilon P} \text{RefCount}(x_i, M) \quad (2)$$

- Consequently, the relative frequency rel(x_i , P) of variable x_i in a population P is the number of references freq(x_i , P) of variable x_i divided by the number of all variable references as given in Equation 3.

$$rel(x_i , P) = \frac{freq(x_i, P)}{\sum_{k=1}^{n} freq(x_k, P)} \quad (3)$$

In our churn prediction problem, finding the most relevant features of the data that affect the classifiers help us in different ways. First, it helps the company to focus on these features, and analyze how they may affect the customers. Moreover, from a machine learning perspective, knowing these relevant features helps in building a fast classifier that can label the new data based on few number of inputs.

5. Experiments and Results

To evaluate the proposed GP-CSL technique and see how effective it is in detecting churns, experiments were performed using 2-folds cross validation, where the data is separated into 50% for training and 50% for testing. Taking into account that our data is imbalanced we applied stratified sampling in splitting the data, in order to preserve the ratio of each class in the training and testing parts. The cross-validation is repeated 10 times, yielding 10 random partitions of the original data, and these 10 results are averaged to produce one estimation. All experiments are conducted using HeuristicLab 3.3.9 framework [23]. A systematic experimentation process was conducted to tune the parameters of GP. For this, different population sizes were experimented (*i.e.*, 50, 100, 150, 200, and 500). For mutation and crossover rates, GP was experimented at 2%, 5%, 10% and 15% for mutation, and 85%, 90%, and 95% for crossover. The best performance was obtained with the parameters values listed in Table 2.

Table 2. GP Parameters

Parameter	Value
Mutation probability	15%
Crossover probability	95%
Population size	50
Maximum generations	100
Selection mechanism	Tournament selector
Elites	1
Operators	+,-,*,\

5.1. Dataset Description

The dataset used in this work was provided by a major cellular telecommunication company located in Jordan. The data set contains 11 attributes of randomly selected 5000 customers subscribed to a prepaid service for a time interval of three months. The attributes cover outgoing/incoming calls related statistics. The data were provided with a class label for each customer indicating whether the customer churned (his subscription is terminated) or still active. The dataset has a highly imbalanced class distribution where the total number of churners is 381, which forms 7.6% of the total number of customers. A list of the variables along with their brief description are given in Table 3.

Table 3. Churn Data Feature Description

Feature	Feature name	Description
F1	3G	Subscriber is provided with 3G service (Yes, No)
F2	Total Consumption	Total monthly fees (calling+sms) in (JD)
F3	Int'l calling fees	Monthly fees for international calling (JD)
F4	Int'l MOU	Total of international outgoing calls in minutes
F5	Int'l sms fees	Monthly fees for international sms (JD)
F6	Int'l sms count	Number of monthly international sms
F7	Local sms fees	Monthly local sms fees (JD)
F8	Local sms count	Number of monthly local sms
F9	Total MOU	Total minutes of use for all outgoing calls
F10	On net MOU	Minutes of use for on-net-outgoing calls
F11	Churn	Churning customer status (yes, No)

5.2. Model Evaluation Metrics

In order to evaluate the developed churn prediction model, we refer to the confusion matrix shown in Table 4, which is the primary source for accuracy estimation in classification problems. Based on this confusion matrix, the following different criteria are used for evaluation:

Table 4. Confusion Matrix

	Actual	
	nonChurners	**Churners**
Predicted nonChurners	**TP**	**FP**
Predicted churners	**FN**	**TN**

1. Accuracy: is the percentage of the total number of predictions that were correctly classified.

$$\text{Accuracy} = \frac{\text{TN}+\text{TP}}{\text{TN}+\text{TP}+\text{FN}+\text{FP}} \quad (4)$$

2. Recall: is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. It can be given by the following equation:

$$\text{Recall\_Churners} = \frac{\text{TP}}{\text{TP+FN}} \quad (5)$$

$$\text{Recall\_nonChurners} = \frac{\text{TN}}{\text{TN+FP}} \quad (6)$$

3. Precision-nonChurners: is the fraction of relevant instances among the retrieved instances. It can be given by the following equation

$$\text{Precision\_nonChurners} = \frac{\text{TN}}{\text{TN+FN}} \quad (7)$$

4. G-mean: is the geometric mean of the recalls of each class and it can be measured by the following equation:

$$\text{G} - \text{Mean} = \sqrt{\text{Recall\_Churners X Recall\_nonChurners}} \quad (8)$$

5.3. Results

GP-CSL was tested using different penalty cost matrices. In each matrix the cost of the churner class is increased by +5, starting from 5 until 30. The following expressions will be used to represent the penalty cost matrices [1:5], [1:10], [1:15], [1:20], [1:25], and [1:30]. However, the traditional fitness function that uses the accuracy metric is the same as using a penalty cost matrix with cost 1, thus we can represent it as [1:1]. Choosing these costs came from extensive experiments with different penalty costs, where choosing 30 as the last penalty cost was because G-Means values decreased significantly with larger costs. The drop in G-mean values reflects the high detection of churners but also high misclassification of normal customers. The results of the proposed GP-CSL using five different cost matrices are compared with three traditional GP fitness functions (accuracy, Pearson R^2, and Mean Squared Error - MSE) and are shown in Table 5. MSE and Pearson R^2 are calculated as shown in Equation 9 and 10, respectively.

$$\text{MSE} = \frac{1}{\text{n}}\sum_{\text{i=1}}^{\text{n}}\left(\text{y}_\text{i} - \hat{y}_\text{i}\right)^2 \quad (9)$$

$$R = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \ \sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}} \quad (10)$$

Where y is the actual value, $\hat{y}$ is the predicted value, $\bar{y}$ is the mean of actual value, $\bar{\hat{y}}$ is the mean of predicted value, and n is the total number of instances. The results of GP using accuracy as a fitness achieved the highest accuracy value compared with other fitness functions and GP-CSL cost matrices. However, this does not mean that this model can perfectly detect churners as the data is imbalanced and this result might represent an example of over-fitting problem.

The recall of Churner and recall of nonChurner results that represent the probability of detection of churners and nonChurners are shown in Table 5. GP using accuracy as a fitness achieved a recall of nonChurner 0.9994 (which is the highest), and a recall of churner equal to 0.7034 (the lowest value compared with other fitness functions). GP-CSL using cost penalty matrix of [1:30] achieved the highest recall value of churner equals to 0.9467, however it achieved the lowest recall of nonChurners 0.6648. The fitness function that got good recall results for both churner and nonChurner was GP-CSL with penalty cost matrix [1:20], which achieved 0.8341 for recall of churners and 0.8493 for recall of nonChurners.

Figure 3 shows the results of recalls for both churners and nonChurners for all GP-CSL fitness functions using different penalty cost matrices. Where [1:1] represents the accuracy fitness function. It is clear from the figure, that the accuracy fitness function detects the nonChurners better than churners due to the data imbalanced feature. Moreover, GP-CSL

using [1:30] penalty cost matrix focuses on churners by giving high cost for not detecting churners, thus results in low recall for nonChurner detection.

The results of accuracy and G-Means are shown in Figure 4. Where it shows that the accuracy drops down when GP-CSL uses higher penalty costs. The accuracy values start with 0.9769 for [1:1] penalty cost matrix, and reach 0.6863 with [1:30] penalty cost matrix. The G-Means results start with 0.8385 with [1:1] penalty cost (accuracy fitness) then get higher with higher costs to reach the highest value of 0.8488 with penalty cost matrix [1:15]. Then it falls down to reach 0.7933 with [1:30] penalty cost matrix. The results of identifying the most relevant features are shown in Table 6 and Figure 5.

In Table 6, the Bold values are larger than 0.1. It can be concluded that "3G" is an important feature for all fitness functions using different cost penalty matrices, "Total-Consumption" has good impact values for 6 out of 7 fitness functions, "Total-MOU" has good impact values for 4 fitness functions, "On-net-MOU" and "Local-sms-fees" impact values were good for 3, and 2 fitness functions respectively. Other features got low impacts (lower than 0.1). As a summary, Features "3G", "Total-Consumption", "Local-sms-fees", "Total-MOU", and "On-net-MOU" are the most relevant features. Figure 5 shows the feature impact values for different penalty cost functions, where it can be inferred that using different penalty cost affects the importance level of the features. For example, "Local-sms-fees" feature had low importance with low penalty cost but got higher importance value when GP used higher cost [1:30]. However, "Total-Consumption" feature had high importance value with [1:1] penalty cost, but decreased significantly with high penalty cost [1:30]. This helps the decision makers to know which feature has a high effect on customers and their decision to stop using this service.

Table 5. Results of Proposed GP-CSL using Different Cost Matrices for Five Metrics (Prec: Precision, Rec: Recall, Ch: Churner, nonCh: nonChurner

Fitness Func.	Accuracy	Prec-nonCh	Rec-Ch	Rec-nonCh	G-Mean
Accuracy	**0.9769**	0.9761	0.7034	**0.9994**	0.8385
Pearson R^2	0.97392	0.9765	0.7097	0.9957	0.8406
MSE	0.7910	0.9795	0.7824	0.7917	0.7871
[1:5]	0.9709	0.9765	0.7105	0.9924	0.8397
[1:10]	0.9407	0.9783	0.7425	0.9570	0.8428
[1:15]	0.8833	0.9827	0.8102	0.8892	**0.8485**
[1:20]	0.8481	0.9841	0.8341	0.8493	0.8413
[1:25]	0.7686	0.9887	0.8950	0.7582	0.8236
[1:30]	0.6863	**0.9936**	**0.9467**	0.6648	0.7922

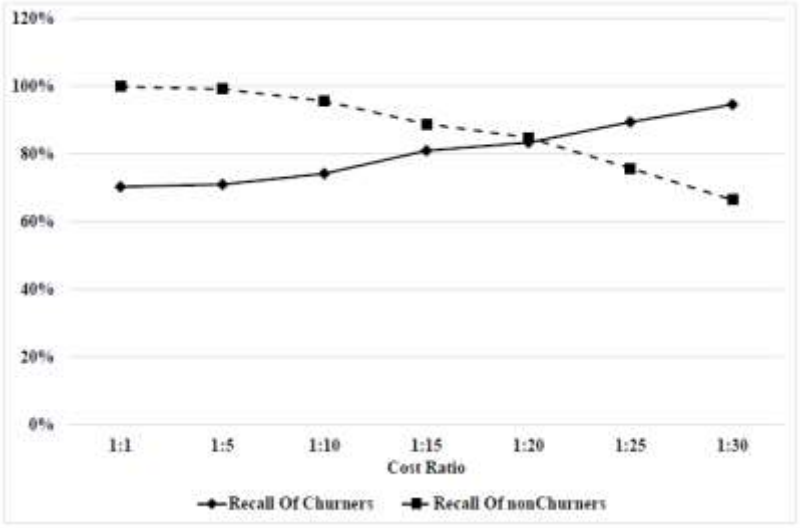

Figure 3. The Results of Recalls for both Churners and nonChurners for all GP-CSL using Different Penalty Cost Matrices

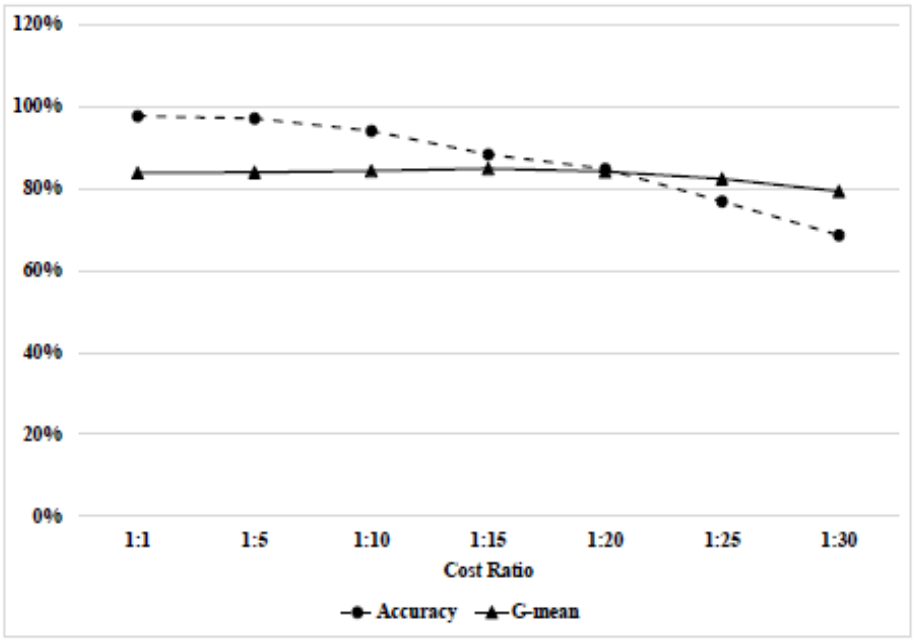

Figure 4. The Results of Accuracy and G-Mean for both Churners and nonChurners for all GP-CSL using Different Penalty Cost Matrices

Table 6. Features Impact after using Different Penalty Costs Matrices with GP-CSL

Cost Matrix	[1:1]	[1:5]	[1:10]	[1:15]	[1:20]	[1:25]	[1:30]
3G	**0.187**	**0.185**	**0.291**	**0.328**	**0.322**	**0.381**	**0.4**
Total-Consumption	**0.295**	**0.281**	**0.118**	**0.209**	**0.161**	**0.168**	0.07
Int'l-calling-fees	0.008	0.02	0.013	0.03	0.048	0.026	0.048

Int'l-MOU	0.013	0.027	0.037	0.042	0.045	0.023	0.015
Int'l-sms-fees	0.022	0.015	0.033	0.039	0.03	0.057	0.062
Int'l-sms-count	0.019	0.016	0.03	0.047	0.025	0.018	0.039
Local-sms-fees	0.016	0.01	0.009	0.022	0.056	**0.223**	**0.26**
Local-sms-count	0.011	0.012	0.025	0.06	0.046	0.029	0.07
Total-MOU	**0.333**	**0.4**	**0.285**	0.087	**0.108**	0.042	0.013
On-net-MOU	0.097	0.034	**0.159**	**0.136**	**0.16**	0.034	0.023

5.4. Comparison with other Classifiers

A comparison of GP using different fitness functions (accuracy, Pearson R^2, MSE, and cost sensitive learning) with other well-known classifiers is conducted and discussed in this section. Other well-known classifiers were implemented using WEKA [24], and HeuristicLab [23]. The comparison was based on five metrics, accuracy, precision, recall-Churner, recall-nonChurner and G-Mean. Other classifiers used in the comparison are: Multilayer Perceptron Neural network classifier (MLP), Neural network ensemble classifier Random forest classifier [25], decision tree classifier (J48), Bayes Network learning (BN) K-nearest neighbours classifier (KNN), and Naive Bayes classifier (NB). The settings used for each classifier are shown in Table 7.

Table 7. Settings of Comparative Classifiers

Classifier	Parameter	Value
MLP	Hidden neurons	12
Random Forest	Number of trees	10
Ensemble of MLPs	Number of networks	10
KNN	Number of neighbors (K)	1

The comparisons results are shown in Table 8, where the average of 10 runs, and the standard deviation (presented in brackets) are reported. As we notice in the table, the proposed GP-CSL outperforms most of the other classifiers in detecting the churner customers. Moreover, other classifiers struggle to handle the detection of churners and nonChurners where they either achieve high detection of churners or nonChurners not both.

For example, MLP achieved 0.9829 for recall of nonChurners, but 0.3360 for recall of churners. Moreover, NB achieved 0.9381 for recall of churners but 0.4390 for recall of nonChurners.

Looking at the G-mean metric, GP-CSL outperforms other classifiers and obtained the higher values. The last three rows in Table 8, show the results of other related work done on the same churn data [12] and [15]. The best accuracy achieved from related work was 0.9713, where GP-Accuracy achieved better accuracy (0.9769) and GP-CSL [1:5] achieved a close value (0.9709). The best RecallChurner achieved by related work is 0.827, where GP-CSL [1:20], GP-CSL [1:25], and GP-CSL [1:30] achieved higher values (0.8341, 0.8950, and 0.9467 respectively). Values of recall-nonChurner and G-mean was not stated in related work, therefore, we could not compare these metrics.

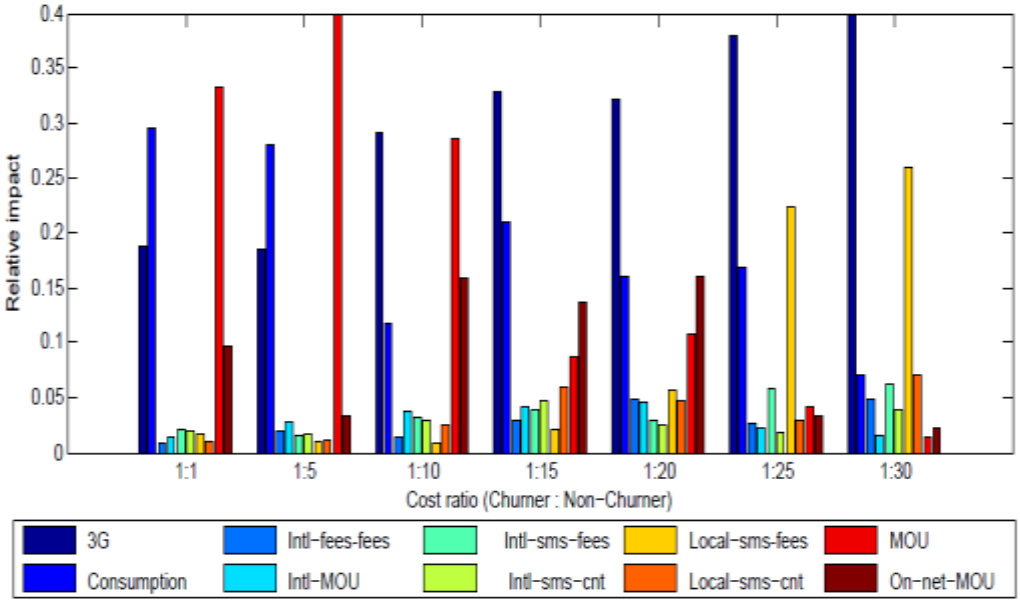

Figure 5. Feature Impact Values for Different Penalty Cost Functions used by GP-CSL

Table 8. Results of other Classifiers Compared with GP using Different Fitness Functions (Prec: Precision, Rec: Recall, Ch: Churner, nonCh: nonChurner)

Classifier	Accuracy	Prec-nonCh	Rec-Ch	Rec-nonCh	G-Mean
	AVE	AVE	AVE	AVE	AVE
	(STD)	(STD)	(STD)	(STD)	(STD)
GP-Accuracy	**0.9769**	0.9761	0.7034	**0.9994**	0.8385
	(0.0010)	(0.0000)	(0.0000)	**(0.0010)**	(0.0000)
GP-Pearson R^2	0.9739	0.9765	0.7097	0.9957	0.8406
	(0.0021)	(0.0006)	(0.0085)	(0.0028)	(0.0049)
GP-MSE	0.7910	0.9795	0.7824	0.7917	0.7870
	(0.1685)	(0.0123)	(0.1645)	(0.1921)	(0.1778)
GP-CSL [1:5]	0.9709	0.9765	0.7105	0.9924	0.8397
	(0.0017)	(0.0001)	(0.0013)	(0.0019)	(0.0005)
GP-CSL [1:10]	0.9407	0.9783	0.7425	0.9570	0.8428
	(0.0157)	(0.0010)	(0.0157)	(0.0180)	(0.0052)
GP-CSL [1:15]	0.8833	0.9827	0.8102	0.8892	**0.8485**
	(0.0198)	(0.0019)	(0.0260)	(0.0235)	**(0.0039)**
GP-CSL [1:20]	0.8481	0.9841	0.8341	0.8493	0.8413
	(0.0248)	(0.0016)	(0.0210)	(0.0285)	(0.0059)
GP-CSL [1:25]	0.7686	0.9887	0.8950	0.7582	0.8236
	(0.0198)	(0.0010)	(0.0126)	(0.0223)	(0.0071)
GP-CSL [1:30]	0.6863	**0.9936**	**0.9467**	0.6648	0.7922
	(0.0454)	**(0.0032)**	**(0.0315)**	(0.0517)	5(0.0170)
MLP	0.9336	0.9472	0.3360	0.9829	0.5746

	(0.0006)	(0.0000)	(0.0000)	(0.0007)	(0.0002)
Random Forest	0.9767	0.9763	0.7058	0.9991	0.8397
	(0.0004)	(0.0001)	(0.0008)	(0.0004)	(0.0004)
MLP (Ensemble)	0.9568	0.9783	0.7373	0.9749	0.8478
	(0.0008)	(0.0001)	(0.0008)	(0.0009)	(0.0005)
J48	0.9754	0.9642	0.7037	0.9978	0.8379
	(0.0790)	(0.0143)	(0.0008)	(0.0009)	(0.0003)
BN	0.9606	0.7482	0.7320	0.9794	0.8467
	(0.4138)	(0.0424)	(0.0053)	(0.0049)	(0.0016)
KNN	0.9594	0.7388	0.7247	0.9788	0.8422
	(0.1974)	(0.0199)	(0.0075)	(0.0023)	(0.0041)
NB	0.4770	0.1213	0.9381	0.4390	0.6416
	(1.5972)	(0.0032)	(0.0043)	(0.0174)	(0.0124)
NCR + CPSO [12]	0.894	0.694	0.827	N/A	N/A
Flat ensemble of ANN [15]	0.958	0.725	0.732	N/A	N/A
Ensemble of NCL-ANN [15]	0.9718	0.814	0.803	N/A	N/A

6. Conclusion

This paper proposed the application of genetic programming using cost sensitive learning (GP-CSL) to solve churn prediction problem as an optimized classification problem. GP-CSL used different penalty costs for prediction errors ranging from 5 to 30 with a step of 5, each is represented in a penalty cost matrix as [1:5], [1:10], [1:15] to [1:30]. A real data from the telecommunication market was used to test the proposed approach. Experiments compared proposed approach with other well-known GP classification fitness functions such as accuracy, Pearson R^2, and Mean Squared Errors. Comparisons were done between all these approaches using different metrics such as accuracy, Precision, Recall, and G-means.

The results showed that the proposed approach GP-CSL achieved better detection of churners using higher penalty cost. Moreover, GP-CSL using a penalty cost matrix [1:20] achieved better detection rates of churners and normal customers at the same time. Comparison with other well-known classifiers, and other related work were conducted also, and results showed that GP-CSL outperform other classifiers and previous work in detecting churners.

Moreover, the paper conducted an analysis of the unique property of GP of finding the most relevant features in the dataset and compared them using different penalty cost matrices and accuracy fitness function. The tests found that out of 10 features 4 to 5 features are the most relevant features used in different fitness functions of GP, and the most relevant features can vary by changing the penalty cost.

Our future plans include testing the same approach on different real datasets from different fields, such as weather detection. Also, we plan to study the effect of the top five most relevant features using other classifiers instead of using the whole dataset features.

References

[1] W. Verbeke, D. Martens, C. Mues and B. Baesens, "Building comprehensible customer churn prediction models with advanced rule induction techniques", Expert Systems with Applications, vol. 38, no. 3, **(2011)**, pp. 2354-2364.

[2] T. Vafeiadis, K. I. Diamantaras, G. Sarigiannidis and K. C. Chatzisavvas, "A comparison of machine learning techniques for customer churn prediction", Simulation Modelling Practice and Theory, vol. 55, **(2015)**, pp. 1-9.
[3] K. W. De Bock and D. Van den Poel, "An empirical evaluation of rotation-based ensemble classifiers for customer churn prediction", Expert Systems with Applications, vol. 38, no. 10, **(2011)**, pp. 12293-12301.
[4] O. Adwan, H. Faris, K. Jaradat, O. Harfoushi and N. Ghatasheh, "Predicting customer churn in telecom industry using multilayer perceptron neural networks: Modeling and analysis", Life Science Journal, vol. 11, no. 3, **(2014)**.
[5] Y. Zhao, B. Li, X. Li, W. Liu and S. Ren, "Customer churn prediction using improved one-class support vector machine", International Conference on Advanced Data Mining and Applications, Springer, **(2005)**, pp. 300-306.
[6] M. Owczarczuk, "Churn models for prepaid customers in the cellular telecommunication industry using large data marts", Expert Systems with Applications, vol. 37, no. 6, **(2010)**, pp. 4710-4712.
[7] H. Faris, B. Al-Shboul and N. Ghatasheh, "A genetic programming based framework for churn prediction in telecommunication industry", Computational Collective Intelligence. Technologies and Applications, vol. 8733 of Lecture Notes in Computer Science, Springer International Publishing, **(2014)**, pp. 353-362.
[8] J. Burez and D. Van den Poel, "Handling class imbalance in customer churn prediction", Expert Systems with Applications, vol. 36, no. 3, **(2009)**, pp. 4626-4636.
[9] B. Al-Shboul, H. Faris and N. Ghatasheh, "Initializing genetic programming using fuzzy clustering and its application in churn prediction in the telecom industry", Malaysian Journal of Computer Science, vol. 28, no. 3, **(2015)**.
[10] Y. Zhang, H. Li, M. Niranjan and P. Rockett, "Applying cost-sensitive multiobjective genetic programming to feature extraction for spam email filtering", Genetic Programming, **(2008)**, pp. 325-336.
[11] J. Li, X. Li and X. Yao, "Cost-sensitive classification with genetic programming", in 2005 IEEE Congress on Evolutionary Computation, vol. 3, **(2005)**, pp. 2114-2121.
[12] H. Faris, "Neighborhood cleaning rules and particle swarm optimization for predicting customer churn behavior in telecom industry", International Journal of Advanced Science and Technology, vol. 68, **(2014)**, pp. 11-22.
[13] A. C. Bahnsen, D. Aouada and B. Ottersten, "A novel cost-sensitive framework for customer churn predictive modeling", Decision Analytics, vol. 2, no. 1, **(2015)**, pp. 1-15.
[14] N. Glady, B. Baesens and C. Croux, "Modeling churn using customer lifetime value", European Journal of Operational Research, vol. 197, no. 1, **(2009)**, pp. 402-411.
[15] A. Rodan, A. Fayyoumi, H. Faris, J. Alsakran and O. Al-Kadi, "Negative correlation learning for customer churn prediction: A comparison study", The Scientific World Journal, **(2015)**.
[16] A. Keramati, R. Jafari-Marandi, M. Aliannejadi, I. Ahmadian, M. Mozaffari and U. Abbasi, "Improved churn prediction in telecommunication industry using data mining techniques", Applied Soft Computing, vol. 24, **(2014)**, pp. 994-1012.
[17] J. Xiao, G. Teng, C. He and B. Zhu, "One-step classifier ensemble model for customer churn prediction with imbalanced class", in Proceedings of the Eighth International Conference on Management Science and Engineering Management, Springer, **(2014)**, pp. 843-854.
[18] A. E. Eiben and J. E. Smith, "Introduction to Evolutionary Computing", SpringerVerlag, **(2003)**.
[19] H. Jabeen and A. R. Baig, "Review of classification using genetic programming", International Journal of Engineering Science and Technology, vol. 2, no. 2, **(2010)**, pp. 94-103.
[20] R. Sivaraj and T. Ravichandran, "A review of selection methods in genetic algorithm", International journal of engineering science and technology, vol. 3, no. 5, **(2011)**, pp. 3792-3797.
[21] C. X. Ling and V. S. Sheng, "Cost-Sensitive Learning", Boston, MA: Springer US, **(2010)**, pp. 231-235.
[22] G. Kronberger, S. Wagner, M. Kommenda, A. Beham, A. Scheibenpflug and M. Affenzeller, "Knowledge discovery through symbolic regression with heuristiclab", in ECML/PKDD (2), **(2012)**, pp. 824-827.
[23] S. Wagner, G. Kronberger, A. Beham, M. Kommenda, A. Scheibenpflug, E. Pitzer, S. Vonolfen, M. Kofler, S. Winkler, V. Dorfer and M. Affenzeller, "Architecture and design of the heuristiclab optimization environment", In Advanced Methods and Applications in Computational Intelligence, Springer, **(2014)**, pp. 197-261.
[24] I. H. Witten, E. Frank, M. A. Hall and C. J. Pal, "Data Mining: Practical machine learning tools and techniques", Morgan Kaufmann, **(2016)**.
[25] L. Breiman, "Random forests", Machine learning, vol. 45, no. 1, **(2001)**, pp. 5-32.

