Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 6, June 2021: 1102-1108

AI Based Nose for Trace of Churn in Assessment of Captive Customers

Drumil Joshi [a*], Aayush Parikh [b], Ritika Mangla [c], Fawzan Sayed[d], Sunil H Karamchandani [e]

[a*,d,e] Dwarkadas J Sanghvi College of Engineering, Mumbai, Maharashtra
[b] Manipal Institute of Technology, Manipal, Karnataka
[c] Veermata Jijabai Technological Institute,Mumbai,Maharashtra

Abstract

Customer churn speculation is aimed at finding customers with a high potential for attraction. Predictive accuracy, Precision, and justification are the three critical elements of a churn predictive model. According to domain knowledge, the Accurate standards of the model allow us to identify the main drivers for customers to churn and develop an effective retention strategy. In this research paper, we present a comparative study of the most popular machine learning classifiers used to solve the problem of churning customers in the telecommunications sector. In the first phase of our test, all models were implemented and tested using statistical evaluative measures on the popular telecom database. In the second phase, the performance improved by boosting was studied. In order to determine the most efficient parameter combinations, we performed hyperparameter tuning for the best classifier and a wide range of parameters. The best overall classifier was XG Boost classifier after Hyperparameter tuning with an accuracy of almost 82% and Precision of 0.8..

Keywords: churn, machine learning classifiers, telecommunication, boosting, hyperparameter tuning, XG Boost

1. Introduction

Amid an increase in competition in the telecommunication field, every company seeks to maximize their customer base. This can be done in two ways: either adding new customers or retaining those who wish to discontinue using the company's service. The cost for retaining old customers is said to be 6-7 times less than persuading and acquiring new customers. Therefore, the companies aim to decrease their potential of customer churn. Customer Churn is a phenomenon where the existing customers stop associating with the company. Depending on the company, they stop using their product or services. A higher customer churn rate would signify that more customers stopped buying the company's service. The ideal customer churn rate would be 0% for any company. Customer churn can be a stumbling block for a rapidly expanding company, so a retention plan should be devised to prevent an escalation in customer churn rates. If done early enough, predicting which customers are likely to leave may help the company promptly implement retention strategies like discounts coupons or offers. Churn prediction models help in identifying such customers. We implemented 8 Machine Learning models, namely Logistic Regression, Random Forest Classifier, Decision Tree Classifier, Support Vector Classifier, AdaBoost Classifier, Gradient Boosting Classifier, XGBoost Algorithm. Next, based on six parameters, we conducted a comparative study to pick the best model. The selected model was then subjected to Hyperparameter Tuning to find an optimal combination of hyperparameters that minimizes a predefined loss function and produces better performance.

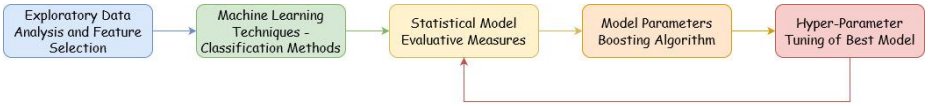
2. Literature Survey

Customer churn prediction has been an area of research where several methods have been explored. These methods mainly use data mining and machine learning. One of the successful techniques was proposed by *M Makhtar et al*. They propose a new classification model based on the Rough Set Theory to classify customer churn. The reason for using RST is that, RST guarantees an efficient and feasible algorithm to find hidden

patterns and rules in data mining. Adnan et al surveyed six well-known sample techniques and compare the performance of these key mechanisms, namely, mega-trend diffusion function (MTDF), a few minority modelling process, a flexible performance modelling model, high-N couples returning to the nearest neighbour k, most of the weight determined process, and immune defence strategies. They concluded that the overall predictive performance of MTDF and rules generation based on genetic algorithms performed the best. Using evaluation metrics like AUC & lift, J. Burez and D. Van den Poel investigated an increase in sample performance (both random and advanced sample) as well as two modelling techniques that increased gradients and weighed less random forests and compared them to other standard modelling techniques..

3. Methodology

Figure 1: Block Diagram of Workflow



3.1 Exploratory Data Analysis and Feature Selection

To find the trends, anomalies and patterns within the data, we use the open-ended process of Exploratory Data Analysis. First, we drop all the rows with null values, and convert the Churn from containing "Yes" and "No" to containing 1 and 0, as this helps in process of machine learning. Using the matplotlib library functions, we then plot the bar graph for the Churn (dependent variable) attribute and its correlation with the other features. We also see the plots for various other features to check for outliers, none of which were found. We see, the plots for how the distribution of Churned and Not Churned people depending on the various columns to get an idea about what parameter influences the results in what way. Scaling of the independent variables is lastly carried out, so the trained model could be much more accurate than before.

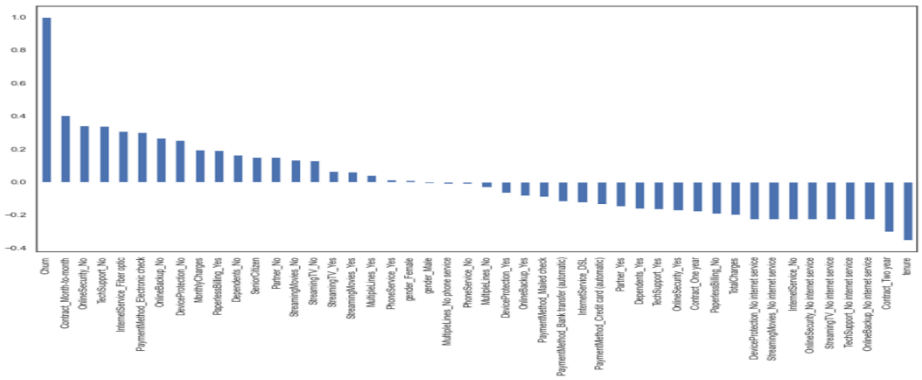


Figure 2: Correlation Bar Plot for Churn

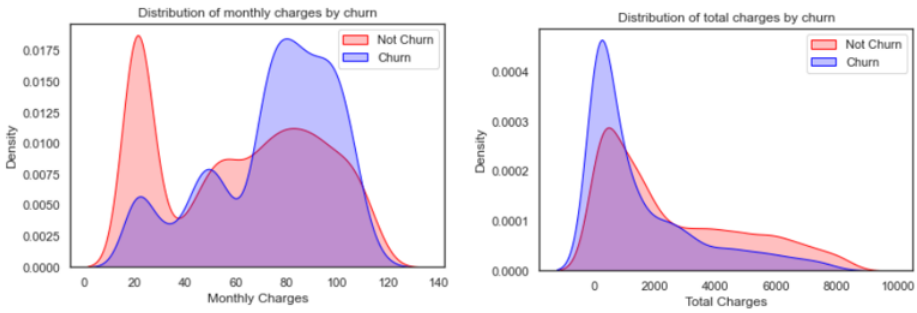


Figure 3: Density Plot for Various Features

3.2 Machine Learning Techniques- Classification Methods

The goal is to develop a Machine Learning model that has high accuracy and interpretability. Using the sklearn library, we divide the data into 70:30 ratio for training and testing respectively. The Machine Learning algorithms considered are as follows:

1. Logistic Regression- As the name suggests, it uses the logistic function or the sigmoid function at the core to the algorithm. An example of a logistic equation is:

$$y = \frac{e^{b_0+b_1\times x}}{1+e^{b_0+b_1\times x}}$$

where y = the predicted output

b_0,b_1 = coefficients that must be learned from the training data

Using this formula, we get an output for a single variable model. The formula can then be changed to taking multiple independent variables as input, only the number of coefficients that need to be trained (b) will increase. The proposed algorithm takes the set of customer dependent parameters which are mapped by a sigmoidal activation function. The proposed model is heavily biased against the client by observing an odds ratio of nearly 4. The probabilistic classifier generated an accuracy of nearly 81%. It indicates that model can assure with 72% probability whether the customer will churn or not in by considering the dependent variables.

2. Random Forest Classifier- In Random Forest, there are multiple decision trees involved in each model and these decision trees work in together in ensemble mode. The individual feature's importance is calculated with the following formula:

$$RFfi_i = \frac{\sum_{j\in all\ trees} normfi_{ij}}{T}$$

where $RFfi_i$ = feature i's importance

$normfi_{ij}$ = normalized feature importance of feature i in tree j

T = total number of trees

Random Forest classifier proves to be resistant to the problem of multi-collinearity and overfitting by averaging or combining the results of different decision trees. It has very less bias-variance trade off. The current classifier greatly discriminates against the churning of client by having an odds ratio less than logistic regression of approximately 3.5. The ensemble classifier was accurate in producing an accuracy of 79% and produced has a precision of 78% of customer churning out. The kappa coefficient measured under different circumstances of customer churning out was approximately 0.46.

3. Decision Tree Classifier- It consists of a tree like structure, where a branch represents a decision node and the leave represent an outcome. Split points are created using the Gini Index:

$$Gini(D) = 1 - \sum_{i=1}^{m} {P_i}^2$$

where P_i = probability that tuple in D belongs to class C_i

The Gini Index is used to determine if a split is needed to be made. Running this throughout the algorithm leads to the development of the entire tree structure. The proposed algorithm takes a set of client-based parameters programmed with a gin launch function. The proposed approach is highly biased towards the client by considering the odds ratio of 2.67. It has been shown that the model can guarantee 64% probability whether a customer will be churned through a variety of dependencies.

4. Support Vector Classifier- It is a classification that is very effective in higher dimensional spaces. The equation is as follows:

$$decision_function = \sum_{i\in SV} y_i\alpha_i K(x_i, x) + b$$

where x = given sample

α_i = dual coefficient

$K(x_i, x)$ = kernel used

b = intercept

A single plane is hence made in our case, which determines if the outcome should be "Churn" or "Not Churn". The proposed algorithm takes a set of client-dependent parameters organized by the non-linear kernel. The proposed approach is highly client-oriented with an odds ratio of approximately 4.3. The probable classifier showed approximately 81% accuracy. It has been shown that the model can guarantee 79% probability of whether a customer will be churn through various dependent features.

5. AdaBoost Classifier- It is a type of meta-estimator classifier. It fits additional copies of the original classifier with some minor adjustments. The updated weight of each training example is:

$$D_{t+1}(i) = \frac{D_t(i)e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

where Z_t = normalization factor

D_t = weight of previous level

x_i, y_i = training set

The term α_t is adjusted for each copy of the classifier, specifically to properly compensate for the weaker features in our dataset. The algorithm fits the weak learners in our dataset onto previous modified version of the data. It gives a comparatively high odds ratio of 4.2 and an accuracy with our dataset of 82%. The model can promise if a customer will be churned with an 80% probability.

6. Gradient Boosting Classifier- Gradient Boosting allows for optimization of loss functions, using an additive model. The algorithm is:

$$F_m(x) = F_{m-1}(x) + \sum_{j=1}^{Jm} \gamma_{jm} R_{jm}$$

where R_{jm} = specific region

γ_{jm} = boosting coefficient

This is applied to each feature, where $F_{m-1}(x)$ represents the weightage of the feature in the previous iteration. The algorithm makes a tree with our given data and applied all the boosting functions to it. The accuracy of the model is a very high 82%. The odds ratio generated is also acceptable at 4.03.

7. XGBoost Classifier- It is also a type of gradient boosting, just one that reduces the computation time greatly with the help of parallel computing and cache awareness. The formula for the objective function (which needs to be minimized) is:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i{}^{(t-1)} + f_t(x_i)\right) + \Omega(f_t)$$

This complex equation is used to apply a similar concept to that of Gradient Boosting, while keeping the state of the computer hardware also in consideration. The odds ratio given by this model is the highest with 4.2. The model guarantees with a precision of 80% that a customer will be churned and the accuracy for the same is also 82%.

3.3 Statistical Evaluative Measures

To evaluate the effectiveness of classifiers in the foresee of customer churn of various schemes by their relevant model parameters, we use statistical measures of accuracy, recall, precision, odds ratio, kappa coefficient and gini coefficient, calculated from the content of the confusion matrix as shown:

Confusion matrix for classifier evaluation.

		Predicted class	
		Churners	Non-churners
Actual class	Churners	TP	FN
	Non-churners	FP	TN

Figure 4: The output disturbed by Outliers

True positive and false positive cases are denoted as TP and FP, respectively, while true negative and false negative cases are denoted as TN and FN.

Accuracy is the total number of predictions that were correct and calculated from the calculation:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Recall is the number of positive cases correctly identified and calculated from the equation:

$$Recall = \frac{TP}{TP + FN}$$

Precision is the number of positive predicted cases that are correct and calculated from the equation:

$$Precision = \frac{TP}{TP + FP}$$

The odds ratio is defined as the probability of success in comparison to the probability of failure and is derived from equation:

$$\log(Odds) = \log(\frac{P}{1 - P})$$

Kappa coefficient compares the accuracy of the system to the accuracy of a random system and is derived from the equation:

$$kappa = \frac{totalAccuracy - randomAccuracy}{1 - randomAccuracy}$$

Gini coefficient is used to compare the quality of various models and to test their predictive ability and is derived from the equation:

$$Gini = 2 \times AUC - 1$$

3.4 Model Parameters Boosting Algorithm

The main aim of boosting is to improve classification performance through the combination of decisions from many classification models, which are called weak machine learning classifiers. In this paper, the effectiveness of boosting on a classifier will be measured by its ability to improve the respective statistical evaluative measures. Boosting algorithm is a combination of weak classifiers used as subroutines. For each weak classifier, the reinforcement algorithm maintains the distribution of weights in the training patterns, so that each pattern contributes differently to the final training error of the classifier. The first popular boosting algorithm is AdaBoost, short for adaptive boosting. It's a meta-algorithm, which can be used in conjunction with many other machine learning algorithms to improve their performance. AdaBoost is flexible in the sense that the following classifiers are structured according to those conditions that were poorly set by previous classifiers. Gradient Boosting Classifier creates additional classification models by inserting a sequential function of the parameter into the current residuals in squares at least in each iteration. It is shown that both the accuracy of the measurement and the performance of the gradient boosting can be significantly improved by random input in the process. Specifically, in each iteration a small sample of training data is drawn randomly from a complete set of training data. This randomly determined sample is used instead of the full sample to fit the student base and calculate the current iteration model update.

XG Boost stands for extreme Gradient Boosting. XG Boost is an implementation of advanced gradient boosting designed for speed and performance. It's a guided and integrated learning approach that involves trees provide a distinction of the type of machine production that is more efficient usually consisting of large numbers of trees as well the ratio of the results of the high forecast. the XG boost classifier makes good results in prediction model but it takes more training time for iteration process. XG boost classifiers handles the sparse dataset in which the missing values are handled properly. XG boost algorithm gives very good training model and performance for prediction of accuracy. The XG boost algorithm addresses one of the key problems in tree learning which is to find the best split.

3.5 Hyper Parameter Tuning for the Best Model

The most powerful ML algorithms are known for capturing patterns and standard data by automatically tuning thousands (or millions) of so-called "readable" parameters. For example, in tree-based models (stems, random forests, XGBoost), readable parameters to select variable options for each number and numerical constraints used to determine whether to take a left or right branch when a prediction is generated. From a coherence point of view, supervised ML descends on a reduction of a certain job loss (e.g., MSE or classification error) based on data entry input data (X_t, Y_t), readable parameters we identify with, and hyper-

params. An important note here is that this reduction is done by allowing only the readable parameters to be different, while holding data and hyper-params regularly. Choosing the perfect set of hyper-params, the most common way to make cross-validation.Given a hyper-param vector h, its quality is assessed by evaluating the loss function on a held-out set of validation data using the best set of learnable parameters,a^*, for that value of h. Symbolically, we have

$$CVLoss(h) \coloneqq Loss(X_v, Y_v; a^*(h), h),$$

$$a^*(h) \coloneqq argmin_a\{Loss(X_t, Y_t, a, h)\}$$

The tuning was done by grid search cv keeping the objective of binary logistic and learning rate of 0.1 with number of estimators of 150 and to define a grid of parameters based on to get the maximum accuracy keeping the number of folds or cross validation set to be 5. After getting the entire grid we again run the XG Boost classifier with the defined parameter grid and to by notice we witness an approximate accuracy increase of 0.5-1%, and it can assure 80% probability whether the customer will be taken in by the dependent variables.

4 Results and Discussions

From the graphs, we can see that decision tree and random forest is the least suitable for our application. They give a low score for all the evaluation parameters compared. Logistic regression and SVM classifier are next in the order. They have a good odds ratio, but are low compared to the others for the rest, especially in respect of precision. The rest of the classifiers, namely Gradient Boosting and Ada Boosting have some of the best results. They have high values for all the parameters and hence fit well for the application. But the best results is given by XGBoost Classifier, giving high results for all the parameters, with no significant compromise in any one of them. It also has reduced computation time, taking a lot shorter to execute.

Figure 4: The output disturbed by Outliers

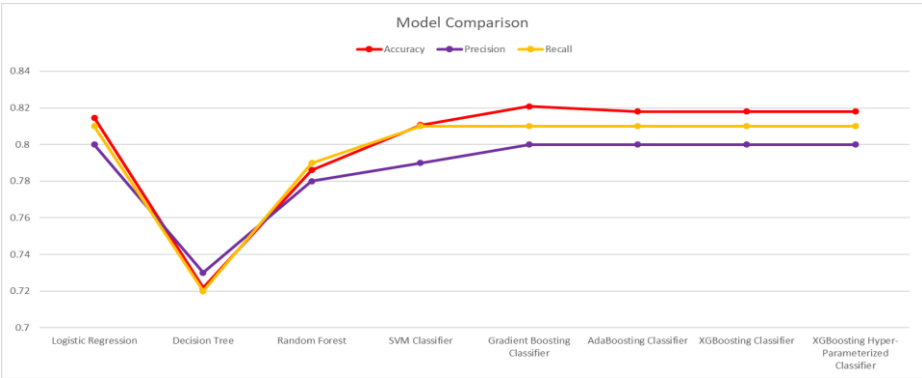


Figure 5: Comparison of Model Performance

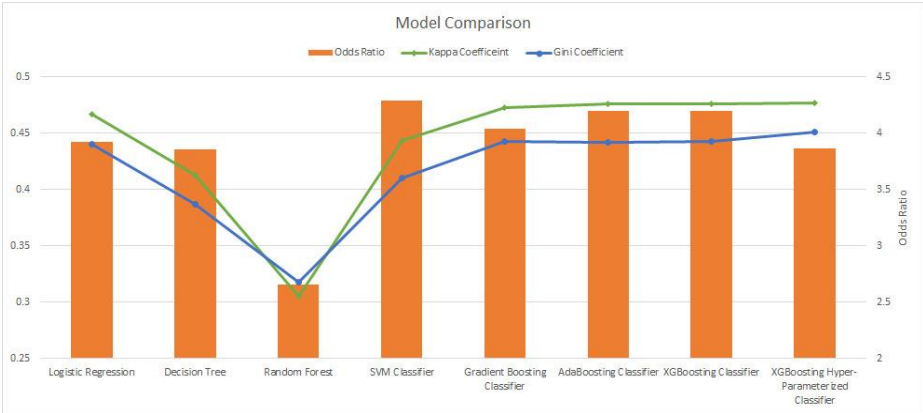


Figure 6: Comparison of Model Performance

5 Conclusion and Future Scope

On comparing the final results, we thereby conclude that the XG Boosted Hyper-Parameterized Classifier proves to be the most vital classifier. It has the highest accuracy of 81.13% and is 0.1-0.3% more as compared to all the remaining classifiers. The Odds Ratio which explains the comparison of a customer churning out or not in the given two different circumstances that is exposure vs. absence of exposure. The XG Boosted Hyper-

Parameterized Classifier has a odds ratio which is 2% less as compared to the other classifiers. Looking at the Kappa coefficient value of 0.477, we see that the concluded classifier have predicted values have Moderate Agreement with the actual values, but still the highest amongst the rest of the classifiers. The marquee point which makes the classifier best suited is that it has a true positive ratio of 0.535 and false positive ratio if 0.096 and that can be depicted seeing the ROC curve. Having the highest Gini coefficient of 0.451 shows that the XG Boosted Hyper-Parameterized Model is the best for the application purposes.

References

[1] Abraham, A., & Ramos, V. (2003). Web usage mining using artificial ant colony clustering. In the congress on evolutionary computation (pp. 1384–1391). IEEE Press.

[2] Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In VLDB (pp. 487–499).

[3] Athanassopoulos, A. (2000). Customer satisfaction cues to support market segmentation and explain switching behavior. Journal of Business Research, 47(3), 191–207.

[4] Amin A, Anwar S, Adnan A, Nawaz M, Howard N, Qadir J, Hawalah A, Hussain A. Comparing oversampling techniques to handle the class imbalance problem: a customer churn prediction case study. IEEE Access. 2016;4:7940–57.

[5] Burez, J., & Van den Poel, D. (2009). Handling class imbalance in customer churn prediction. Expert Systems with Applications, 36(3), 4626–4636. doi:10.1016/j.eswa.2008.05.027

[6] Coussement, K., & Van den Poel, D. (2008). Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert Systems with Applications, 34, 313–327.

[7] Cohen, W. W. (1995). Fast effective rule induction. In A. Prieditis & S. Russell (Eds.), Proceedings of the 12th international conference on machine learning (pp. 115–123). Tahoe City, CA: Morgan Kaufmann.

[8] Martens, D., Van Gestel, T., & Baesens, B. (2009). Decompositional rule extraction from support vector machines by active learning. IEEE Transactions on Knowledge and Data Engineering, 21(2), 178–191

[9] Suykens, J., Van Gestel, T., De Brabanter, J., De Moor, B., & Vandewalle, J. (2002). Least squares support vector machines. Singapore: World Scientific

[10] Witten, I. H., & Frank, E. (2000a). Data mining: Practical machine learning tools and techniques with Java implementations. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc