Churn  Prediction in Telecommunication Industry

ashish.akshantal

## Purpose

The high increase in the number of companies competing in mature markets makes customer retention an important factor for any company to survive. Thus, many methodologies (e.g., data mining and statistics) have been proposed to analyze and study customer retention. The validity of such methods is not yet proved though. This paper tries to fill this gap by empirically comparing several techniques: Customer churn - logistic regression models, Random forest, Gradient boosting machine and Deep neural network. The paper proves the superiority of logistic regression technique and stresses the needs for more advanced methods to churn modelling.

Let’s consider a typical engagement journey in a Telecom Company.



**The Problem**

It is generally too late to take retention actions after a consumer churns out. Success rates on retaining users are around 2–3%.

Now with **Churn Prediction**, there is one small but impactful difference



Now, you can take whatever actions you were going to take after churn beforehand and thereby improve success rates of retaining users to **up to 70%.**

## So what is Churn really?

For subscription companies it’s pretty simple, users who stop paying for the product, they are considered churned users. For non-contractual businesses, if a user does not complete a **critical event** on the platform within a fixed **window** of time, then customer is considered churned from the platform.

The window is determined by how often you expect users to perform the critical events. For example, businesses which require frequent usage such as Games have windows as small as 1 Day and when it comes to Food Delivery, it can be a week and where the critical event is ordering food.

## Why is Churn so Important to worry about?

### 1. Funding just gets harder as you keep going forward

Churn is one of the main Key Performance Indicator (KPI’s) which good investment funds use to evaluate companies and it becomes more important as the company grows because as you raise further, you need to prove that people actually want to use your product. Moreover, the market becomes saturated so keeping customers you have already acquired becomes important.

### 2. Increased Costs and Lost Revenue

Acquiring a new Customer is usually 5–25 times more expensive than retaining one. 5 % increase in Churn can decrease Profits by 25–95%

### 3. The Leaky Bucket problem in Growth:

A Company’s net customer growth will remain stagnated if don’t they manage Churn. Losing the same amount of already acquired users as you bring in new users is just not scalable.

### 4. Customer Dissatisfaction:

If customers are not engaged with the product, that means there is likely they’ll talk about your product and there is less chance for Companies to get Viral growth. This would mean that the Customer Acquisition Cost would still be high.

## Churn Modelling Techniques

### Logistic Regression

Logistic regression is a data mining technique used to predict occurrence probability of customer churn. Logistic regression is based on a mathematically-oriented approach to analyse the affecting of variables on the others. Prediction is made by forming a set of equations connecting input values (i.e., affecting customer churn) with the output field (probability of churn). The equations (1), (2) and (3) give the mathematical formulas for a logistic regression model (Miner, Nisbet, & Elder, 2009).

𝑝(𝑦 = 1 𝑥1 , … , 𝑥𝑛) = 𝑓(𝑦) (1)

𝑓(𝑦) = 1 (1+𝑒−𝑦 ) (2)

𝑦 = 𝛽0 +𝛽1 𝑥1 + 𝛽2 𝑥2 + ⋯ + 𝛽𝑛 𝑥𝑛 (3)

Where:

• y is the target variable for each individual j (customer in churn modelling), y is a binary class label (0 or 1);

•  𝛽0 is a constant;

•  𝛽1, 𝛽2 ….. 𝛽𝑛 is the weight given to the specific variable associated with each customer j (j=1,…. m);

•  𝑥1 , 𝑥2 , … 𝑥𝑛 , are the predictor variables for each customer j, from which y is to be predicted.

Customer data sets are analysed to form the regression equations. An evaluation process for each customer in the data set is then performed. A customer is can be at risk of churn if the p-value for the customer is greater than a predefined value (e.g. 0.5),

Multicollinearity resulted from strong correlations between independent variables is a concerning factor in logistic regression models as well . The existence of strong multicollinearity leads to incorrect conclusions about relationships between independent and dependent variables since it inflates the variances of the parameter estimates and gives wrong magnitude of the regression coefficient. Under certain circumstances, logistic regression can be used to approximate and represent nonlinear systems in spite of it is a linear approach.

### Decision Tree

Decision trees, a popular predictive models, is a tree graph presenting the variables’ relationships . Used to solve classification and prediction problems, decision tree models are represented and evaluated in a top-down way. The two phases to develop decision trees are tree building and tree pruning. Starting from the root node representing a feature to be classified, decision tree is built. Selecting a feature can be done by evaluating its information gain ratio or gini index. The lower level nodes are then constructed in similar way to the divide and conquer strategy. Improving predictive accuracy and reducing complexity, pruning process is applied on decision trees to produce a smaller tree and guarantee a better generalization by removing branches containing the largest estimated error rate. The decision about a given case regarding to which of the two classes it belongs is thus made by moving from the root node to all leaves. Though there are many algorithms for building decision tree, CART, C5.0 and CHAID are those most used.

###

### Decision trees have several advantages.

First, they are easy to visualize and understand .

Second, no prior assumptions about the data are needed since it is a nonparametric approach.

Third, decision trees can process numerical and categorical data.

### On the other hand, decision trees suffer from some disadvantages.

First, its performance is affected by complex interactions among variables and attributes.

Second, complex decision trees are very hard to be visualised and interpreted.

Third, it suffers from the lack of robustness and over-sensitivity to training data sets .

### Random Forest

Random forests or random decision forests are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_%28statistics%29) of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees’ habit of [overfitting](https://en.wikipedia.org/wiki/Overfitting) to their [training set](https://en.wikipedia.org/wiki/Test_set)

When the training set for the current tree is drawn by sampling with replacement, about one-third of the cases are left out of the sample. This [oob (out-of-bag) data](https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr) is used to get a running unbiased estimate of the classification error as trees are added to the forest. It is also used to get estimates of variable importance.

After each tree is built, all of the data are run down the tree, and [proximities](https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#prox) are computed for each pair of cases. If two cases occupy the same terminal node, their proximity is increased by one. At the end of the run, the proximities are normalized by dividing by the number of trees. Proximities are used in replacing missing data, locating outliers, and producing illuminating low-dimensional views of the data.

The out-of-bag (oob) error estimate

In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows:

Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree.

Put each case left out in the construction of the kth tree down the kth tree to get a classification. In this way, a test set classification is obtained for each case in about one-third of the trees. At the end of the run, take j to be the class that got most of the votes every time case n was oob. The proportion of times that j is not equal to the true class of n averaged over all cases is the oob error estimate. This has proven to be unbiased in many tests.

Variable importance

In every tree grown in the forest, put down the oob cases and count the number of votes cast for the correct class. Now randomly permute the values of variable m in the oob cases and put these cases down the tree. Subtract the number of votes for the correct class in the variable-m-permuted oob data from the number of votes for the correct class in the untouched oob data. The average of this number over all trees in the forest is the raw importance score for variable m.

If the values of this score from tree to tree are independent, then the standard error can be computed by a standard computation. The correlations of these scores between trees have been computed for a number of data sets and proved to be quite low, therefore we compute standard errors in the classical way, divide the raw score by its standard error to get a z-score, ands assign a significance level to the z-score assuming normality.

If the number of variables is very large, forests can be run once with all the variables, then run again using only the most important variables from the first run.

For each case, consider all the trees for which it is oob. Subtract the percentage of votes for the correct class in the variable-m-permuted oob data from the percentage of votes for the correct class in the untouched oob data. This is the local importance score for variable m for this case, and is used in the graphics program [RAFT](https://www.stat.berkeley.edu/~breiman/RandomForests/cc_graphics.htm).

Gini importance

Every time a split of a node is made on variable m the gini impurity criterion for the two descendent nodes is less than the parent node. Adding up the gini decreases for each individual variable over all trees in the forest gives a fast variable importance that is often very consistent with the permutation importance measure.

### Gradient Boosting Machine

Gradient boosting is a [machine learning](https://en.wikipedia.org/wiki/Machine_learning) technique for [regression](https://en.wikipedia.org/wiki/Regression_%28machine_learning%29) and [classification](https://en.wikipedia.org/wiki/Classification_%28machine_learning%29) problems, which produces a prediction model in the form of an [ensemble](https://en.wikipedia.org/wiki/Ensemble_learning) of weak prediction models, typically [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning). It builds the model in a stage-wise fashion like other [boosting](https://en.wikipedia.org/wiki/Boosting_%28meta-algorithm%29) methods do, and it generalizes them by allowing optimization of an arbitrary [differentiable](https://en.wikipedia.org/wiki/Differentiable_function) [loss function](https://en.wikipedia.org/wiki/Loss_function).

### How Gradient Boosting Works

Gradient boosting involves three elements:

1.      A loss function to be optimized.

2.      A weak learner to make predictions.

3.      An additive model to add weak learners to minimize the loss function.

### 1. Loss Function

The loss function used depends on the type of problem being solved. It must be differentiable, but many standard loss functions are supported and you can define your own.

For example, regression may use a squared error and classification may use logarithmic loss.

A benefit of the gradient boosting framework is that a new boosting algorithm does not have to be derived for each loss function that may want to be used, instead, it is a generic enough framework that any differentiable loss function can be used.

### 2. Weak Learner

Decision trees are used as the weak learner in gradient boosting. Specifically regression trees are used that output real values for splits and whose output can be added together, allowing subsequent models outputs to be added and “correct” the residuals in the predictions.

Trees are constructed in a greedy manner, choosing the best split points based on purity scores like Gini or to minimize the loss.

Initially, such as in the case of AdaBoost, very short decision trees were used that only had a single split, called a decision stump. Larger trees can be used generally with 4-to-8 levels. It is common to constrain the weak learners in specific ways, such as a maximum number of layers, nodes, splits or leaf nodes.

This is to ensure that the learners remain weak, but can still be constructed in a greedy manner.

### 3. Additive Model

Trees are added one at a time, and existing trees in the model are not changed. A gradient descent procedure is used to minimize the loss when adding trees.

Traditionally, gradient descent is used to minimize a set of parameters, such as the coefficients in a regression equation or weights in a neural network. After calculating error or loss, the weights are updated to minimize that error. Instead of parameters, we have weak learner sub-models or more specifically decision trees. After calculating the loss, to perform the gradient descent procedure, we must add a tree to the model that reduces the loss (i.e. follow the gradient). We do this by parameterizing the tree, then modify the parameters of the tree and move in the right direction by (reducing the residual loss).

Generally this approach is called functional gradient descent or gradient descent with functions.

### Performance Evaluation Metrics

The critical issue in using different churn modelling methods relate to: (a) Efficiently assessing the performance of these methods; and (b), bench marking and comparing the relative performance among competing models. This section discusses those issues. 3.1 Classification Accuracy The confusion matrix is a tool that can be used to measure the performance of binary classification mode (also called contingency table). A confusion matrix is a visual representation of information about actual and predicted classifications produced by a classification model.

Table 1 depicts a confusion matrix for a binary classifier.

                                                                               Predicted classes

                                                                          Class=Yes/+/ Churn                    Class=No/-/ No-churn

 Actual classes     Class=Yes/+/ Churn         TP (true positive)                       FN (false negative)

                              Class=No/-/No-churn        FP (false positive)                     TN (true negative)

Different accuracy metrics resulted from the confusion matrix are classification accuracy, sensitivity and specificity. Classification accuracy (CA) is the percentage of the observations that were correctly classified, which can be calculated from the matrix using equation

CA= 𝑇𝑃+𝑇𝑁 /𝑇𝑃+𝐹𝑃+𝑇𝑁+𝐹𝑁

Sensitivity and Specificity

Overcoming some of the weakness of accuracy metric, sensitivity and specificity are used. Sensitivity is the proportion of actual positives that are correctly identified. Specificity is the proportion of actual negatives that are correctly identified. The equations are thus:

Sensitivity= 𝑇𝑃/ 𝑇𝑃+𝐹𝑁

Specificity= 𝑇𝑁 /𝑇𝑁+𝐹𝑃

Receiver Operating Characteristic Curve (ROC)

The Receiver Operating Characteristic (ROC) curve is a depiction of the relations between the true positive rate (i.e., benefits) and false positive rate (i.e., costs), drawn on x and y axis in l leaner scale. ROC represents the relations between the churners ratio correctly predicted as churners, and non-churners ratio wrongly predicted as churners. The ROC provides relative compromises between benefits and costs. The ROC curve consists of points corresponding to prediction results.



# Data

Demographics Data:

•           HouseholdID : Each Household id

•           Country: Country.

•           State: State

•           Retired: Whether retired

•           Has Partner: Demographic information - whether the customer has partner

             (1-Yes; 2-No)

•           Has Dependents: Demographic information - whether the customer has

             dependents (1-Yes; 2-No)

•           Education: Education qualification

•           Gender: Demographic information – gender

Account Information:

•           CustomerID: CustomerID

•           BaseCharges: Customer account information (Charges for Base plan)

•           DOC: Date of data collection

•           TotalCharges: Customer account information (Total).

•           DOE: Date of entry as customer

•           ElectronicBilling: Customer account information - whether electronic billing

•           ContractType: Contract type

•           PaymentMethod: payment method

Data of ServicedOptedFor :

•           CustomerID: CustomerID

•           Type Of Service: Service signed for

•           Service Details:

Churn Data:

•           CustomerID: Customer ID

•           Churn: Whether the customer churns (Target)

## Data Preprocessing

Data preparation phase produces the final data sets for models. The main goal of data preparation is to enhance data quality and improve data analysis performance. Data preprocessing needs to be carried out in a more iterative manner until a conclusive outcome is reached. In this study, data preprocessing includes

> Data aggregation - Used dcast from reshape package to combine the train\_services opted for data with the remaining data.

> Imputation of the missing values - Imputed the missing values using Amelia

> Type Conversions - Converted the attributes to respective data types.

> Feature selection of the most informative variables - Used Variable Importance from the Random Forest model.

> New variable derivation - Derived No\_of\_days ( Difference between DOE and DOC),

> Handling Class Imbalance - Used SMOTE and ROSE for balancing the class distribution

## Exploratory Analysis



## Results

Imputation with Amelia and balancing class variables with SMOTING

1. Logistic Regression -  Recall - 85.7%

Omitting missing values and Imbalanced class distribution

1.  Logistic Regression - Recall - 82, Accuracy - 75



2. Random Forest - Recall - 82, Accuracy -73



3. Gradient Boosting Machines - Recall - 71, Accuracy - 78.57

Omitting missing values and Handling class imbalance using ROSE

1. Logistic regression - Recall - 76, Accuracy - 77



2. Random Forest - Recall - 76, Accuracy - 73

3. Gradient Boosting Machine- Recall - 70, Accuracy - 74

Patterns extraction from decision trees (C5.O)

* Using the same preprocessed data which has been used above, built a decision tree algorithm using C50 to build a classifier.

From the summary of the above model, rules have been extracted which give the information about the

a)Lift

b)Confidence

c)Support

d) Condition

# Analysis

The experimental results conducted in this study show that the Logistic regression has outperformed all the other models. By the result one can assume that the data is linearly separable and hence logistic regression is giving best results.