Churn prediction of subscription user for a music streaming service

Sravya Nimmagadda, Akshay Subramaniam, Man Long Wong

December 16, 2017

This project focuses on building an algorithm that predicts whether a subscription user will churn after their subscription expires using classification models. The models are trained and tested using data provided KKBox, Asia's leading music streaming service's through Kaggle for the WSDM 2018 challenge.

1 Introduction

For subscription business like music, games, magazines etc., accurately predicting the churn (whether a user cancels after the subscription expires) is critical to the long-term success of the company. And even slight variations in churn can drastically affect profits making it very important to predict it accurately so that they can develop strategies based on this analysis.

In this project, we work on building algorithms to predict the churn rate of KKBOX's users. KKBOX is Asia's leading music streaming service, holding the world's most comprehensive Asia-Pop music library with over 30 million tracks. They offer a generous, unlimited version of their service to millions of people, supported by advertising and paid subscriptions. This delicate model is dependent on accurately predicting churn of their paid users. We use the data provided by Kaggle (https://www.kaggle.com/c/kkbox-churn-prediction-challenge) for the analysis and building algorithms.

2 Data Visualization and Feature Engineering

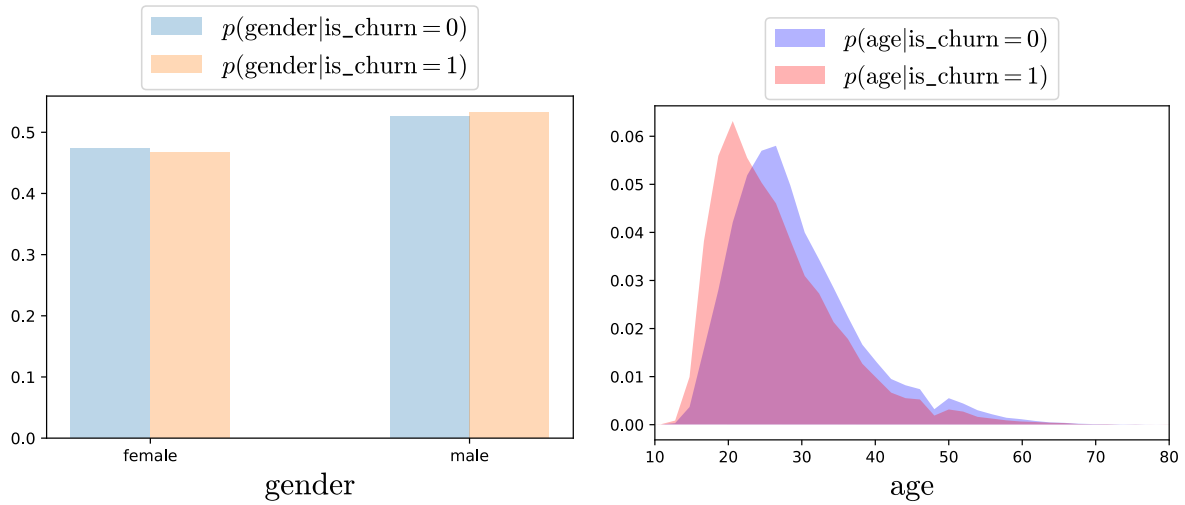
A major part of the work for this project was feature engineering. During the feature engineering process, in total 66 features were selected. These include some raw features in the original datasets such as age, city, payment method in the last transaction, etc. We also derived some additional features, like if the user had any discount, and some historical statistics for the logs.

We used a Naive-Bayes like approach to compute conditional probability densities and used that to inform our choice of features. Figure 1a shows that overall, the probability of the user being male is almost as same as that being female given that the user churn so gender is not included as a feature to predict where the user will churn or not. Figure 1b shows that younger people are more likely to churn so age is included as one of the features.

We also paid close attention to normalizing data and not using any future data in our training process. All dates were normalized by the last day of the previous month and all user logs were normalized by the total number of logs so as to not bias data for users with long histories.

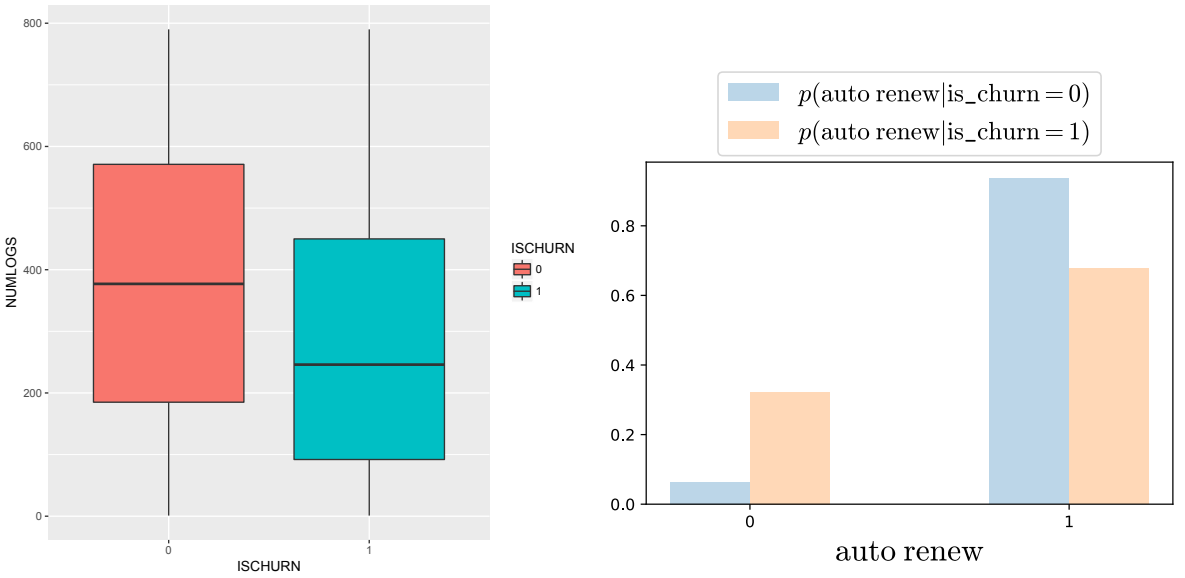
3 Models and methodology

For this binary classification problem, we use the aggregated stats of the user logs of the subscribers, their description and transaction logs as features. The features from user logs include



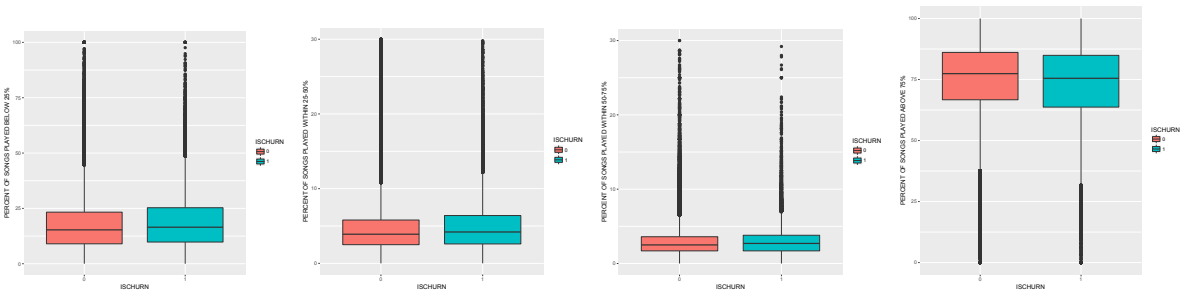
(a) Gender of the subscriber **(b)** Age of the subscriber

Figure 1: Probability mass function of gender and probability density function of age



(a) Number of times logged by a subscriber **(b)** Auto renewal

Figure 2: Number of times logged and auto renewal



(a) $< 25\%$ of the songs played **(b)** $25-50\%$ of the songs played **(c)** $50-75\%$ of the songs played **(d)** $> 75\%$ of the songs played

Figure 3: Percentage of songs played below a given fraction

details like number of logged sessions, fraction of songs played less than 25 percent, fraction of songs played between 25-50 percent, fraction of songs played between 50-75 percent and fraction of songs played above 75 percent. Figures 2, 3 show the churn and not churn characteristics like mean, 25,75 percentiles with respect to the different features mentioned before that is used for the current classification model.

For such an unbalanced dataset, even naively predicting that no user will churn has a 93% accuracy and better metrics than just accuracy should be used to judge the performance of an algorithm. So, instead we use area under the precision-recall curve and more importantly, the recall at 50% precision as the performance metric for this problem. Larger area under the curve is the characteristic of a good and robust classifier.

We start a preliminary analysis and feature engineering using a simple logistic regression model. Based on the results obtained after training a logistic regression model for different set of features, we found that:

1. Normalizing features in an unbiased way was very important. This is absolutely necessary for features like the transaction date. It is also necessary for some derived features. An example is to use the number of minutes logged by a user per day instead of the cumulative number of minutes so as to not bias the data in favor of users that have a long transaction history.

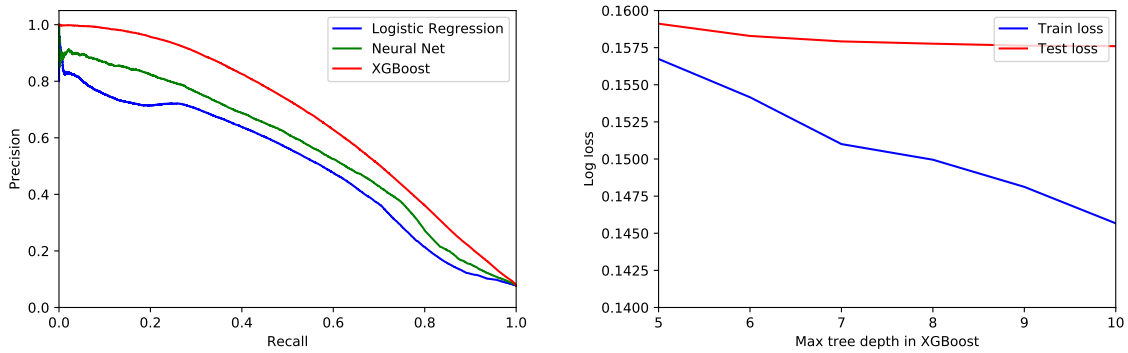
2. It is much more efficient to include intuitive combination of features to create a new set of features than to rely on the model to figure out those dependencies. For example, we obtained better results when we created a boolean feature labeling a transaction as being discounted or not when compared to just training the model based on the payment plan price and the actual payment as separate features.

3. Quantity of data is more important than quality of data. A lot of users had transaction data but no user logs or member data. We found that the model performs much better when we fill in missing data by mean values of different features as opposed to dropping that example.

After the feature engineering process based on logistic regression, we used the final set of features and data to train a neural network with 3 hidden layers and a gradient boosting algorithm (XGBoost) for better classification. Finally, we also tuned the XGBoost model through a hyper-parameter search to get the best results for the available dataset. The train-test split used for the analysis is $80 - 20\%$ on a total data of around 1.8 million examples. We perform stratified splitting to ensure no bias in the train or test data maintaining the 8% share of the churned subscribers. As the full data set is highly skewed, we also try and train the models on a balanced under sampled data set and analyzed its performance with all the three classifiers. More details on the exact training and test set sizes and different models trained are listed in the Table 1.

4 Results

We deployed three different binary classification models to predict whether users will cancel their subscription in the next month. We used the libraries scikit-learn, tensorflow and XGBoost to implement logistic regression, neural network and gradient boosted trees respectively. The accuracies of the three different methods were compared and XGBoost had the best prediction performance based on log loss and recall metrics. The predictions from the gradient boosted trees model gave us a cross-entropy loss of 0.117 on unseen data and we ranked 35 out of 469 teams on Kaggle (as of 12-12-2017).



(a) Precision-recall curve for different classifiers **(b)** Tuning of XGBoost

Figure 4: Performance of different classifiers and tuning of XGBoost

	Size		Log Loss		Test
Model	Train	Test	Train	Test	Recall
XGBoost (Full dataset)	1.57M	0.39M	0.15	0.16	0.67
XGBoost (Balanced dataset)	0.24M	0.39M	0.36	0.37	0.69
Neural Net (Full dataset)	1.57M	0.39M	0.17	0.17	0.63
Neural Net (Balanced dataset)	0.24M	0.39M	0.44	0.43	0.59
Logistic Regression (20% of full dataset)	0.31M	0.08M	0.18	0.19	0.58
Logistic Regression (Balanced dataset)	0.24M	0.39M	0.45	0.43	0.6

Table 1: Performance of different classifiers trained on the full and balance datasets. The reported test recall is computed for a precision of 0.5 on the precision-recall curve.

5 Conclusion and future work

In conclusion, we tested three standard methods for binary classification on the problem of predicting users who will churn from a subscription service. We paid special attention to curating the data in an unbiased way to improve our prediction accuracy. One curious observation was that quantity of data seemed extremely important even if that meant sacrificing some quality of the data. Based on the models we tested, the gradient boosting algorithm performed the better than logistic regression and neural networks.

To improve our predictions, one possible way is to go through the logs and generate churn labels for past months. This way, we can increase our dataset size by an order of magnitude and potentially get much better predictions. Another option would be to use a kernelized SVM classifier. We didn't pursue this approach since our dataset is large and training is prohibitively expensive for > 200000 examples.