Predicting Customer Churn Based on User Data

Background

The business problem to be solved in this project is how to accurately predict user churn for the KKBox subscription service. Churn refers to when a customer has discontinued a subscription service. For a subscription based business to be successful it's churn rate has to lower that its rate of new customer subscriptions. KKBox is a music streaming app in Taiwan and surrounding regions. KKBox presented this as Kaggle competition and provided user data to see if contestants could come up with a better model than what they were using internally.

Specifically, they want to forecast if a user makes a new service subscription transaction within 30 days after the current membership expiration date. Predicting the future is never easy but with such a large sample of user data it should be possible to use a machine learning algorithm to classify likely churners and likely non-churners using insights from their user data. The evaluation metric they set for their challenge is the log loss of the predicted probability of churn for all the users in the test set. An accurate model will help the business identify customers who are likely to churn. This is valuable to the marketing team for applying targeted customer retention efforts, and to analyze what factors are causing customers to churn.

Description Of Data

The data provided by KKBox consists of 5 csv files labeled: train, members, transactions, users logs and sample submission. The train file has two columns, the anonymized user id, and a binary 'is churn' value. There are about 993 thousand rows/users in the train file and they were all selected as having memberships that expired in February 2017. All other information about the users is jumbled somewhere in the members, transactions and user logs files. The sample submission file (evaluation set) is 970,000 rows/user who have been selected for having subscriptions that expire in March 2017. So correctly stated, the train set is for user churn in March 2017, and the test set if for user churn in April 2017.

The members file has almost 6.8 million rows/users and 5 columns of information about each anonymized user id. These columns are city, age, gender, registration method and date of initial registration. About 4.5 million of these users have NaN values in the gender column. There are also about 4.5 million users with a value of zero in the age column and well as about 6,000 outliers that make for nonsensical ages such as -5 or 873.

The transactions file does not have one row per unique user. It is a 21.5 million row by 9 column csv file that has about 2.4 unique users. Each row is a specific transaction that one user made and since KKbox subscription typically renew monthly there is a transaction row for each month the user has been a member. Columns of the transaction file include: payment method id, payment plan days, plan list price, actual amount paid, ‘is auto renew’ binary values, transaction date, membership expire date, and binary ‘is cancel’ values.

Each row of the user logs file contains data about one day of one user’s usage of the KKBox app, so it turns out to be a massive file that is difficult to work with. Even the transactions file can present memory problems at times at a size of 1.6 GB, so the user logs file at size of 28.4 GB certainly requires special treatment. The simplest work-around is to just take the first X million rows of the user logs file and work with that data. The information in the columns of the users log file includes counts of songs listened to grouped by percentage of song played, number of unique songs played, and total seconds played for that day.

So the scope of the user information available both seems somewhat limited but also overwhelming in quantity and shape. Perhaps KKbox has other data available about its users such as what variety of music users listened to, but they did not present that and it would be proprietary so my analysis will be limited to these datasets. Instead of looking for other data I focused on organizing what I have and subsetting it into a usable size dataframe.

An approach that I took was to focus the user log data instead of the transaction data. The rationale being that it should be easier to discern differences in the listening behavior of churners vs non-churners and that listening behavior would have more predictive power. I tried cleaning and wrangling the data in a variety of ways and the latest iteration went as follows in the next paragraph.

First, I read the members csv and apply a boolean filter to exclude members with nonsensical ages. Then merged the members and train set with an inner join on the user id column to create a dataframe called df. Then I read the first 40 million rows of the user logs csv and selected only the entries for users that existed in df. I changed the string values in the date column of the user log into datetime objects so I could filter out specific date ranges of logs. Then as a shorthand way to limit memory usage I saved that abridged version of the user logs as a csv that could be opened in a new notebook with a fresh kernel.

Being that this is a real world problem with real data, I had to wrangle my data and construct my model in a manner that accounts for the fact that there is some amount of incorrect and or missing data. So although initially I tried to filter out users with nonsensical ages, it turned out that would be mean filtering out the majority of users. About half of all users in the dataset had a listed age of 0. This is a common problem with companies that utilize user data from sign up info or email surveys.

In order to convert the daily user logs into a form that can be used with some of the sklearn machine learning algorithms, ie. one row for each user with multitude of feature columns, I used the group-by method on each data column, grouping by the user-id and using the mean as the aggregation function. So this turned the daily user behavior into the average daily user behavior for whatever time period I filter for. I then took the 'mean user logs' then did another inner merge on user id with df. Since this resulting dataframe contained a few column of categorical data I used the pandas 'get dummies' function to convert then into additional columns with binary values. The dummy variables also dealt well with instances incomplete categorical data. For example if there was an NaN value in the gender column, it was replaced by a zero and then became the 'gender_0' binary dummy variable.

It became clear that the transactions data would be necessary to create meaningful features. The way that I ultimately dealt with this datafile was first to create one feature for the number of transactions on file a user has. Then I took the most recent transaction that each user had and added each column of that transaction as a feature for that user.

Since the totality of the user data was too large to process and model using the methods taught in this course and the memory constraints of my machine I settled on using a sample of the user logs data which was about 25% of the total records and filtering out by date to look at the user behavior in just a 2 month period before churn window. The rationale being that churners and nonchurners would be likely to behave differently leading up to their decision on whether to continue their service. This average behavior over the last two months is not an ideal approach from a feature engineering standpoint, but should be enough to extrapolate out a significant model.

Model Selection and Performance

Then came feeding my wrangled dataset to a machine learning algorithm. Since the goal was binary classification I thought from the beginning that this would call for logistic regression. Additionally, since the evaluation metric KKbox chose was the log loss this again suited logistic regression since a logistic regression fit is essentially a probability curve. However I knew that random forest classifiers also tend to work very well with binary classifiers since with only two classes, the branching and computational strain does not become too extreme.

My default logistic regression model scored on the holdout set with a ROC AUC of 0.66 and log loss of 0.244. On the competition submission scoring set its log loss was 0.160. Bear in my mind that the log loss of just randomly guessing would be 0.693 and an absolutely perfect model (no such thing) would be 0.0. Next I tried a random forest with 100 classifiers and got very high ROC AUC of 0.986 and a log loss of 0.09 on the holdout set. However, on the competition scoring set it had a log loss of 0.307.

So this was clearly an example of overfitting since it did extremely well on the train set but not so well on the test set. Next I tried using a cross validated grid search on logistic regression to attempt to find the optimal hyperparameters. Again this resulted in some overfitting where the model had ROC AUC and log loss of 0.93 and 0.134 on the hold out set but log loss of .0270 on the test set.

One thing the grid search did highlight though is how using the ‘L1’ as the choice of the penalty parameter can improve fitting. These parameters deal with how the loss term is calculated with L1 referring to least absolute shrinkage and selection operator (LASSO) regression and L2 referring to ridge regression. I remembered a technique briefly described in this course where LASSO regression can be used for feature selection. An aspect of LASSO is it shrinks the coefficients of certain variable toward zero. So what I did is use LASSO regression then just filtered out the variables with significant magnitude coefficients and then ran ridge regression on just those variables. This resulted in model that had ROC AUC and log loss of 0.88 and 0.174 on the holdout set and a log loss of 0.137 on the test set. This was my best model thus far and I decided this would be the final one I use for the purposes of this project.

Conclusion

The model I constructed can be used by the hypothetical client KKbox to identify users with a high probability to churn. This would be of great use to marketing team for deploying targeting marketing campaigns to increase customer retention, which is one of the primary goals of any subscription based business. Additional analysis of which features relate to customer churn could prove quite valuable for implementing product improvement measures.