Keeping customers

Predict churn and create a retention strategy
April, 2018

Agenda

1. Problem Overview and Goals
2. Summary of Findings to Date
3. Data
4. Approach (models)
5. Tuning, calibrating, measuring success
6. Considerations for productization
7. Your Input!

Problem Overview and Goals

- Our business
    - A subscription based music service
- Problem
    - Lack of insight into which customers will churn, at what volume, and at what profit
- Goals
    - Create a model to predict customer churn from usage and transaction data
    - Create an economic model for retention
    - Recommend a process for keeping the churn and economic retention models updated with latest information

Summary of Findings to Date

Promising results with minimal tuning!

- Primary Prediction Metric: Recall = 78%
  (of users who leave, model correctly predicts 78% of them)
- Secondary Metric: Accuracy = 97% (vs. 94% baseline)
- Best model: XGBoost
- More to come:
  - More data, 50/50 split
  - New model features
  - Model tuning
  - Probability of churn for each user
  - Economic impact for each user

AO

Data

Log data (transactions and user logs) spans 26 months, from Jan. 2015 to Feb. 2017

Available raw data has 4 tables:

Transactions	User Logs	Members	Train
Transaction data for each user. Each row is a payment transaction.	Who, how, and when users used the service. Each row is a unique user-date combination.	Demographic data on each user. Each row represents a unique user.	**Labels of which users churned.** Each row represents a unique user.
21.5M rows X 9 columns 1.6GB	392M rows X 9 columns 29.1GB	6.8M rows X 6 columns 0.4GB	1.0M rows X 2 columns 45MB
Msno: User ID **Payment_method_id**: Payment Method **Payment_plan_days**: Length of plan **Plan_list_price**: Price for the plan **Actual_amount_paid**: Amount paid **Is_auto_renew**: T/F flag determining whether membership is auto-renew or not **Transaction Date**: Date of purchase **Membership_expire_date**: Expiry date **Is_cancel**: T/F flag determining whether or not the user canceled service	**Msno**: User ID **Date**: Date of the logged activity **Num_25**: Number of songs played < 25% of song length **Num_50**: Number of songs played between 25% and 50% **Num_75**: Number of songs played between 50% and 75% **Num_985**: Number of songs played between 75% and 98.5% **Num_100**: Number of songs played between 98.5% and 100% **Num_unq**: Number of unique songs played **Total_secs**: Total seconds played	**Msno**: User ID **City**: City of the user **BD**: Age of the user **Gender** **Registered_via**: Registration method **Registration_init_time**: Initial time of registration **Expiration_date**: Expiration of membership	**Msno**: User ID **Is_churn**: T/F flag variable we are trying to predict.

Approach Overview

- Prepare data for analysis
- Feature engineering
- Build and test models for predicting churn
- Build and test economic model for retention
- Build a flowchart for keeping the models updated

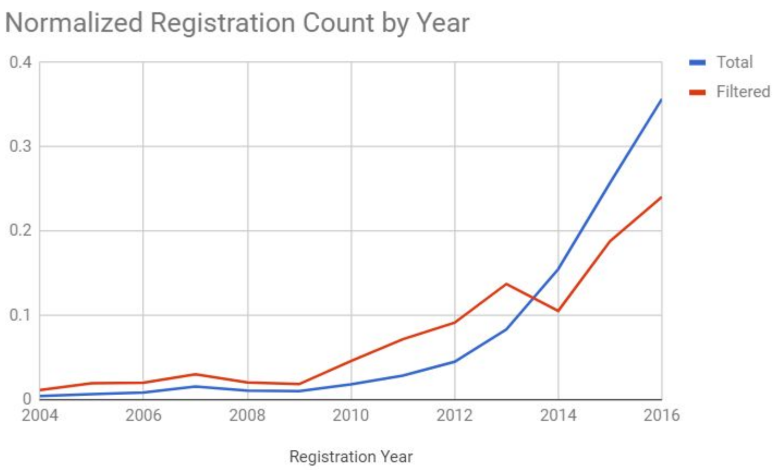
***2017 not included due to partial dataset (through Jan-2017)

AO

Prepare data for analysis

Original Data

- 4 Tables: Transactions, Members, User Logs, Labels
    - Original format: csv
    - Total Size: 31.14 GB
    - 1.02 million labeled users (subset of total business dataset)
- Use Google BigQuery to Filter
    - Select random 1% of members in labels table
    - Inner join with other tables to generate dataset for analysis ensure each user has complete data during analysis
- Export BQ into spreadsheet for model analysis
- Future Work:
    - Return 50/50 split of churn/no-churn
    - Further explore data integrity



	Percent Churn
Total Data	6.56
Filtered Data	5.70

Current Challenges

- Ensure reduced data mimics total dataset
- Handling N/A (gender column has 50%)
- Some outliers (ex age = -7000)

CK

Feature Engineering

Completed to Date

- As-is Features (Members)
  - Demographic: City, Gender, Age, Registration Method
- Transactions
  - Latest transaction information: plan days, amount paid, auto renew, transaction date, expiry
  - Spend: Cost paid/day
- User Logs
  - Min date, max date, tenure
  - Count, sum and mean of songs played 25%, 50%, 75%, 98.5%, 100%, and seconds played for the last 7, 14, 30, 90, 180, and 365 days.

Future Features

- Comparing time frames of user activity and transactions (e.g., do we see payments with minimal usage?)
- Calculate time until expiration date
- Transaction trends (last few transactions)
- User log trends (e.g., has usage notably declined in past week, 2 weeks, 1 month?)
- Better date handling

Models for churn

Model Building

- Train / Dev / Test Split: 60% / 25% / 15%
- Standard sklearn libraries
- Straightforward fit and predict
- Minimal tuning to date ➔ future opportunity! We manually tuned:
  - Regularization parameter for XGBoost
  - Weight parameter to account for 6% / 94% positive / negative split

Measuring Success

Primary Metric: Recall of Users Who Churn

- Definition: Of users who leave, what % does model correctly predict? Currently 78% with XGBoost.
- Enemy #1: Predicting users will stay when they actually leave (false negatives)
- Balance with minimizing predicting users leave who actually stay (false positive rate)

Secondary Metric: Overall Model Accuracy

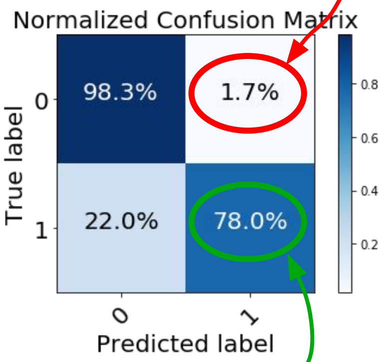
- Definition: Correct predictions / total predictions
- Baseline = 94% (6% of users churn ➔ guessing all 0's)
- Current best = 97.1% with XGBoost.

Current Performance

Model	Recall	Accuracy
XGBoost	78%	97.1%
Random Forest	55%	96.4%
SVM	0%	94.0%
KNN	0%	94.0%
Gaussian NB	N/A	6.0%

XGBoost Conf. Matrix

Minimize False Positive Rate



Maximize Recall

Future Modeling Tasks

Anticipating improved performance from next steps:

- More data (10K → 100K users)
- Balanced split of churn/stay for training data (6/94 → ~50/50)
    - 6% * 10K * 60% = **360 churn users** → 50% * 100K * 60% = **30,000 churn users**
- New model features
- Model tuning (grid search)

Next Step: Calculating Probability of Churn

- For use in economic model (next slide)
- Important we calibrate probabilities (post hoc test to verify model calculates accurately)
- If initial model calibrates poorly, considering hierarchical model
    - **Model 1** (e.g., XGBoost) gives predictions
    - For users Model 1 predicts as positive, **Model 2** (e.g., random forest) calculates probabilities

Abundant pipeline model for retention

- Customers are easy to get and cost $CAC (Customer acquisition cost) to attract
- Our business model says that a great customer generates $OLTV (Optimum lifetime value) of profits
- The value of any customer can be described as $LTV(Lifetime value)/$OLTV
- There's always a risk of these customers leaving us (Churn). We want to spend $RAC(Reacquisition cost) to keep them with us. This could be loyalty programs, targeted marketing, discounts etc
- $RAC has to be less than $CAC as our pipeline is robust and we can always get new customers by spending $CAC
- Value of every customer is different. $RAC should depend on value of the customer ($LTV/$OLTV)
- Also there's a different chance for different classes of customers to leave. $RAC should depend on the chance of that customer leaving
- Combining all of that gives us the following:

$$RAC = .75 * CAC * POC * (LTV/OLTV)$$

Considerations for Productization

Offline Model → **Offline model + Feedback loop** → **Pipeline, Streaming Environment**

Offline Model

Current development on 100k Members

- Focus on recall metric
- Utilize 50/50 split of churn/no-churn
- Calibrate model probabilities with actual churn

Offline model + Feedback loop

Environment utilizing full dataset from music subscription company

- Calibrate model probabilities with actual churn
- Economic model to identify customers and max resources to allocate to prevent churn
- Feedback on impact of retention methods
- Re-calibration and tuning every Quarter

Pipeline, Streaming Environment

Development of streamed ML model

- Data pipeline for transaction and user logs
- Monitor churn in real time, predict
- Create notification and reports for retention/sales

Your Input!

What did you like?

What else would you recommend?

Lousy pipeline model for retention

- Customers are impossible to get and cost $CAC (Customer acquisition cost) to attract
- Our business model says that a great customer generates $OLTV (Optimum lifetime value) of profits
- The value of any customer can be described as $LTV(Lifetime value)/$OLTV
- There's a risk of these customers leaving (Churn) and we losing $LTV in profits
- So spending anything less than $LTV to keep the customer is fair game. This becomes $RAC (Reacquisition cost)
- $RAC as no correlation to $CAC is in this case as $CAC is almost equal to $LTV
- Value of every customer is still different. $RAC should depend on value of the customer ($LTV/$OLTV)
- Also there's a different chance for different classes of customers to leave. $RAC should depend on the chance of that customer leaving
- Combining all of that gives us the following:

$$RAC = .30 * LTV * POC$$

Economic model for retention

$$RAC = .75 * CAC * POC * (LTV/OLTV)$$

- **RAC (Reacquisition cost) is spent to achieve OLTV (Optimum lifetime Value)**
  - RAC is the spend to avoid churn
  - It can include targeted retention marketing, loyalty program costs etc
  - It does not include the regular service and awareness marketing expenses
  - OLTV is the 3 year margin from a customer per our business plan
- **RAC is proportional to POC (probability of churn), LTV (lifetime Value) and CAC (Customer acquisition cost)**
  - LTV is calculated/projected from available transaction information
  - LTV = OLTV is the best case for our business plan
  - LTV/OLTV gives us an idea of the customer value
  - POC is an output from our churn prediction models
  - We're assuming CAC. We'll also keep RAC < CAC
  - Higher LTV can afford a higher RAC
  - Higher POC (Higher risk of churn) needs a higher RAC