The document details out the process of analyzing data in a B2C context, applying domain knowledge for superior feature engineering and building advanced machine learning algorithms to predict churn of a subscription user

User Churn Prediction

KKBOX Music Streaming Subscription

Gaurav Dembla

Feb 28, 2020

Table of Contents

Introduction

The goal of this project is to build an algorithm that predicts whether a user will churn after his/her subscription expires. The binary classification models are trained and tested on sample data sourced from Kaggle. The data was uploaded/provided on Kaggle by the firm KKBOX for the WSDM 2018 challenge.

Context

KKBOX is Taiwan'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 business model is dependent on accurately predicting churn of their paid users.

Based on the available data on existing and former customers (users who terminated the subscription in past), KKBOX would like to predict whether an existing user might churn after his/her subscription expired. Based on the project results, the firm's marketing and/or customer experience teams might intervene to gauge better understanding of the situation by opening communication channels with such likely-to-churn users and assimilating their feedback. After conducting causal analysis on users' feedback and understanding their pain-points, the teams could rectify the situation by taking appropriate tactical and/or strategic actions.

Source Data

The following datasets were available on the Kaggle website. Combined, they turned out to be rich in both length (size) and breadth (attributes) and provided enough records for both model training and testing.

a) **Labeled** data (aka 'train' data in the source repository), denoting churn behavior comprises of unique ids (named as 'msno') of users and their renewal or churn outcome during the month of Mar'17. The 'churn' attribute is the target variable that needs to be predicted. Size/shape: 1mn x 2.

b) **Registration** data (aka 'membership' data in the source repository), containing basic service membership and demographic information about users, such as gender, age and registration method/source. The dataset has membership information for only 88% of users in the training dataset. Size/shape: 7mn x 6.

c) **Sales transaction** data, consisting of information about subscription transactions – whether a transaction was renewal or cancellation of service, amount paid if renewal, whether auto-renewal option was chosen by the user, and the length of membership plan selected. Size/shape: 22mn x 9.

d) **Usage logs**, describing daily listening behavior of a user, such as number of songs played once, replayed, skipped or completed, and the amount of time songs played during the day. Size/shape: 400mn x 9. The dataset has usage information for only 87% of users in the labeled set. It is fair to assume that the users who are missing from this dataset did not log into the music service at all during the time period of data capture (Jan'15 until Feb'17).

Success Criteria

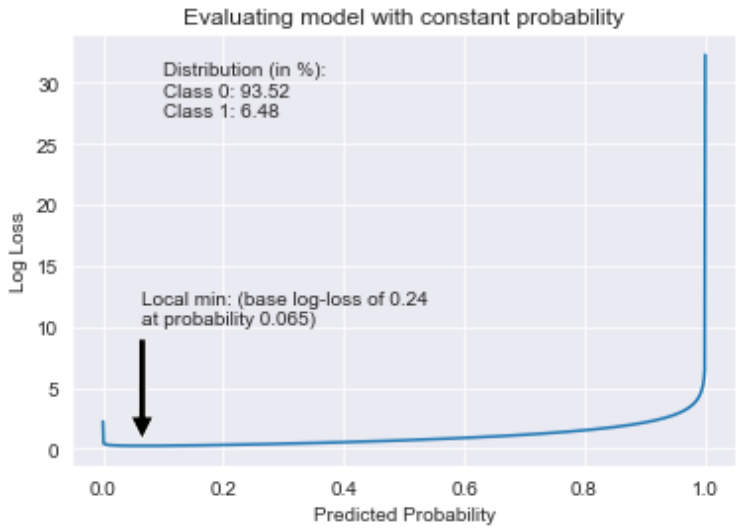
Log-loss score on unseen data (hold-out set) shall be used to assess and compare alternative trained statistical models with each other. Log-loss is indicative of how close probability predictions are to the corresponding actual/true values of 0 or 1.

$$Logloss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \, ln \, p_i + (1 - y_i) \, ln(1 - p_i)]$$

where N is the number of observations, ln is the natural logarithm, y_i is the binary target, and p_i is the predicted probability that y_i equals 1.

A model with lower log-loss value is better than the one with higher log-loss value, provided both the models are applied to the same distribution of dataset. As a result, the model with *lowest* log-loss score on hold-out set will be selected as the final model for real-life predictions.

Additionally, the chosen statistical model should have a log-loss score lower than the **baseline log-loss** score for the given dataset. Base log-loss score is determined by the naïve classification model, which simply pegs all the observations with a constant probability equal to % of data with class 1 observations. As shown in the chart below, in case of the generated hold-out dataset with 6.5% churn users (refer to **Data Extraction** under Data Wrangling section), log-loss score of 0.24 at constant probability of 0.065 is regarded as the base log-loss score.


Evaluating model with constant probability
Distribution (in %):
Class 0: 93.52
Class 1: 6.48
Local min: (base log-loss of 0.24
at probability 0.065)
Log Loss
Predicted Probability
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

Data Wrangling

Data Extraction

Right off the bat, one would notice that usage data is too large to be handled by even an above-average personal computer. Due to limitation of computing and memory resources on the available local machine, a limited sample of 150k users was *randomly* selected from 1mn available in the labeled dataset, bringing down the size of associated raw usage data to a manageable set of 37mn records. Since not all the sample users have registration and usage logs (see below stats), only a subset of users (131k) was retained for exploratory data analysis (EDA) – users for whom all types of information is available in source datasets.

	Labeled	Registration	Sales	Usage
# records	150,000	132,507	2,401,222	37,112,323
# users	150,000	132,507	150,000	131,428

Upon asking and answering myself some interesting questions during exploratory data analysis, discovery of few interesting features through usage data encouraged me to scale down the size of sample users to just 128k for model training and testing – users who have been using the service for at least last eight weeks (refer to **Recent Usage** in Exploratory Data Analysis section).

While 128k users provides enough data records to train the machine learning models on, a hold-out test set of 64k users has been *separately* and *randomly* sourced from the original labeled dataset in a similar manner, ensuring no overlap of training with hold-out set. The hold-out dataset is later used to assess the performance of varied algorithms on *unseen* data.

Sourcing of users from the original labeled dataset in a "random" fashion ensured retention of original distribution of churn and renew users in both training and hold-out sets. Hence, percentage of churn users is ~6.5% of total users in each of the three datasets – original labeled, training and hold-out.

Data Cleaning

Multiple data quality checks were performed to ensure good quality data for further exploratory data analysis and statistical model training. Here is the list of issues discovered as part of data quality assessment, along with solutions, wherever applicable. It is worth mentioning right away that due to reasons mentioned inline, none of the attributes with data quality issue was used for model training.

Missing/null values –

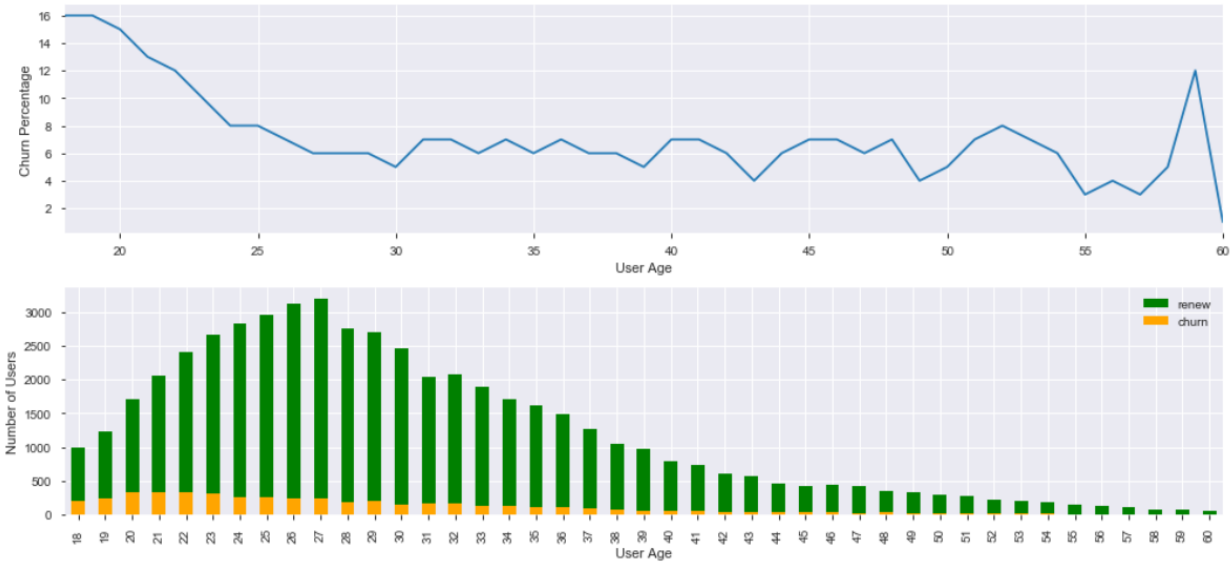
1) **Gender values missing**: Out of 132k sample users available in membership/registration data, more than half of the users (70k) were missing values to discern whether they were male or female. Running a quick analysis on the other half showed that the churn behavior of either of the genders was same (~8%). Since the attribute, with good values, did not show any importance in predicting churn behavior, it was dropped from further analysis and feature engineering. Moreover, imputation was not a viable option as substantial chunk of data was missing for the attribute.

Unreasonable values -

2) **List price less than paid amount**: Out of 2.4mn sample sales (subscription) transactions in total, roughly 100k records showed that the list price of the service plan was less than the amount paid by the user. The data quality issue was fixed by re-assigning list price as paid amount if the former was less than the paid amount in case of renewal transactions. In case of cancellation transactions, list price was not touched, since the attribute probably didn't mean anything for such transactions. After fixing the bad values in all the sales transactions, **discount amount** was derived on just the last transaction for each user using the list price and paid amount attributes. Exploratory data analysis on the new attribute revealed that meagre 300 odd users (out of 150k available in the sample) availed non-zero discount in their last transaction. As a result, the attribute was dropped from model training.

3) **Age of users outside reasonable range**: Out of 132k sample users available in membership/registration data, more than half of the users (75k) were outside a reasonable range of 18-60, with 73k users of age 0. Imputation is not a viable option as substantial chunk of data is missing for the attribute.

   Running a quick analysis on the age group 18-60 reveals that younger users (age<=22) show higher churn rate than older ones whose churn behavior hovers around 8%. Building a restricted model for users within 18-60 age group could show whether the attribute provides unique information for superior classification. If the feature turned out to be useful, two final models could be built - one with age attribute in the feature set and the other one without it.



Given that few records (rare though) show age as low as -51 and as high as 1043, it is reasonable to assume that age is a user-entered field and that no data quality restraints were applied during its capture, posing a big question in front of me – Could I rely on its accuracy for even age group 18-60? Only restricted model training and subsequent feature importance assessment could reveal the answer to this question. However, the attribute has been removed from further model building; perhaps, it could be explored and analyzed in subsequent iterations of the project (refer to Future Work section).

4) **Playtime beyond conceivable range**: Out of 37mn sample usage records, mere 18k show that total time user listened to songs on a given day were beyond minimum and maximum number of seconds in a day – 0 and 86,400 respectively. Even though negligible percentage of total usage records has DQ issue, the effect on the mean value was significant – mean of playtime values within normal range was 8,037 seconds, while mean of all the values (including inconceivable ones) was -306,050 seconds.

   Here are few ways of correcting the issue. However, I decided to use a hybrid methodology of first two, wherein I applied second where average value of the week existed and first if it didn't.

   a) Floor and ceiling: Update negative values as 0, and ones beyond maximum possible as 86,400.

   b) Average of the week: Rectify the issue by replacing an abnormal value with the average value of the week the date belongs to.

   c) Average of the same days: Replace an abnormal value with the average value of the same days across user's lifetime. Such a rectification is based on a reasonable theory that a user's listening behavior must be more consistent with same days of different weeks rather than with other

days of the same or different weeks. For example, my listening behavior on a given Sunday of a given week might be more consistent with other Sundays of different weeks rather than with Mondays or Thursdays of the same week or other weeks.

d) Extrapolate using total number of songs completed: A quick analysis reveals that normal playtime value for a given day is highly correlated with total number of songs completed by the user on the same day, with Pearson correlation coefficient of 0.98. Thereby, abnormal values could be fixed by extrapolating numbers (running regression) based on total number of songs listened to on a day.

Discovery of playtime's high correlation with total number of songs completed prompted me to skip the playtime attribute all together from further analysis and feature engineering, leveraging total number of songs completed in its stead.

Data Manipulation

While registration data is at the user level, both usage and transaction datasets are at user and date level. In order to train a statistical model to predict churn of a user, all the relevant pieces of information has been aggregated at user level and then denormalized/joined together to yield one single feature-set that goes as input to statistical models for training and testing.

Additionally, multiple features have been engineered at the raw level or at the aggregated level to enable superior classification results. Instead of outlining such features here itself and breaking the suspense, they have been deliberately reserved for Exploratory Data Analysis section to bring about an engaging narration in the report.

Exploratory Data Analysis

Last Sales Transaction

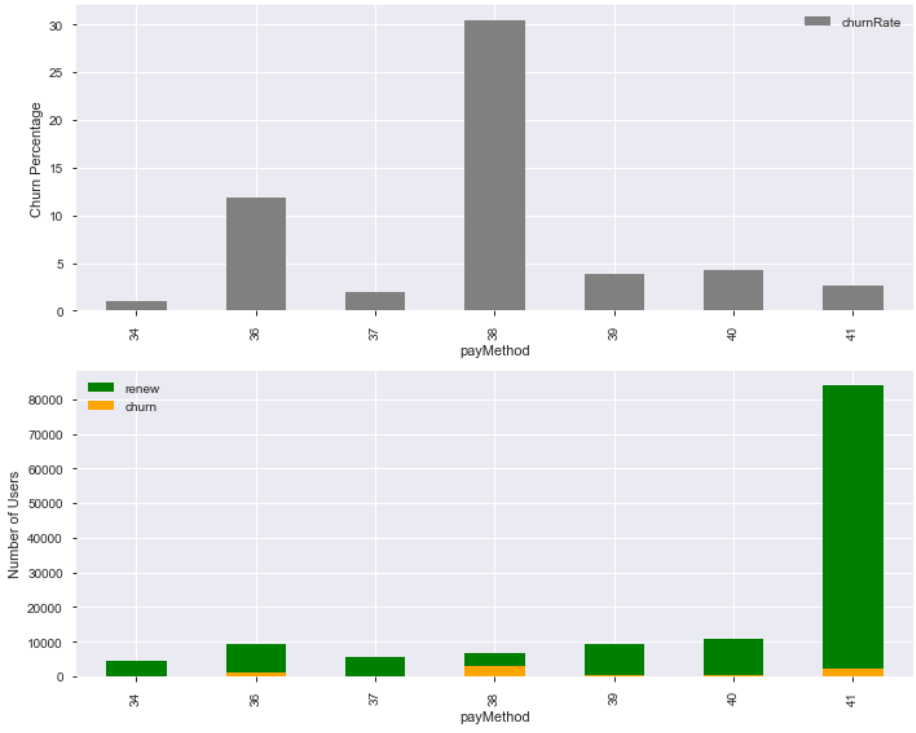
Let's focus on the *latest* sales transaction of each user to find insights into his churn/renewal behavior.

1. **Auto-renew and Cancellation**: Putting together cancellation, auto-renewal and churn information from the last subscription transaction yields some immediate conclusions.

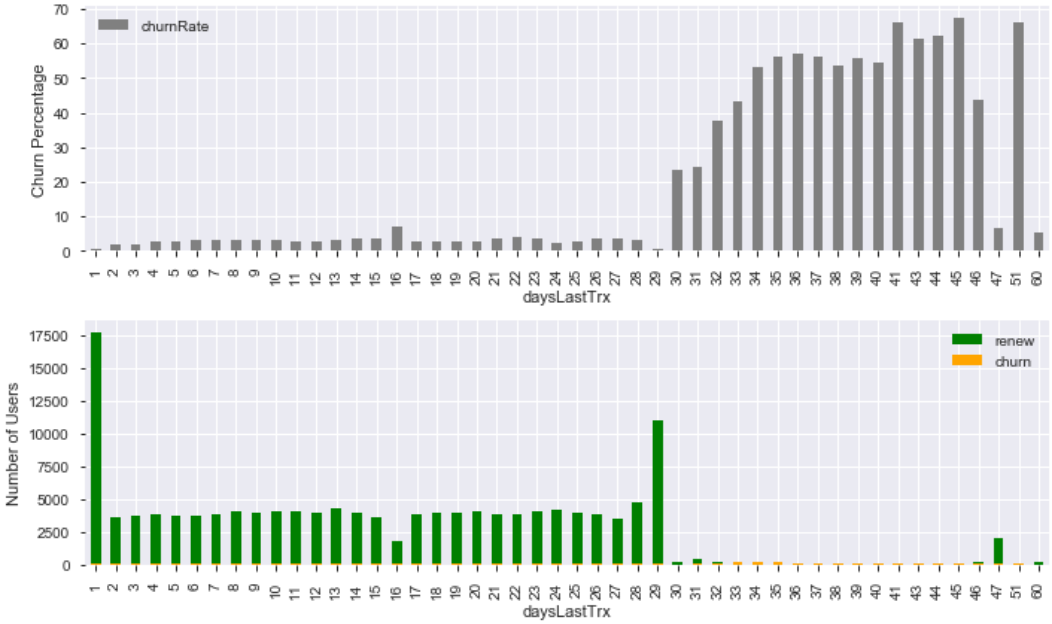
		Churn		
cancel	**autoRenew**	No	Yes	**churnRate**
No	No	11,229	5,549	33.1%
No	Yes	128,559	475	0.40%
Yes	Yes	565	3,623	86.5%
			150,000	

   a) If the user is not on an auto-renewal plan, there is roughly 33% chance that the user will churn.
   b) If the user is on an auto-renewal plan that has not been cancelled yet, he is most likely to continue with the service (~0.4% chance of churning) and let the service plan renew automatically next month (Mar' 2017).
   c) If the user cancelled an auto-renewal plan, he is highly likely to leave the service and not come back to renew it the next month (~86% chance of churning).

2. **Payment method**: Out of ~30 payment methods, analyzing top seven that contributes to over 90% of the total sample users, reveals an interesting fact – payment method 38 shows particularly higher churn rate than others, though it harbors only 6% of total users.



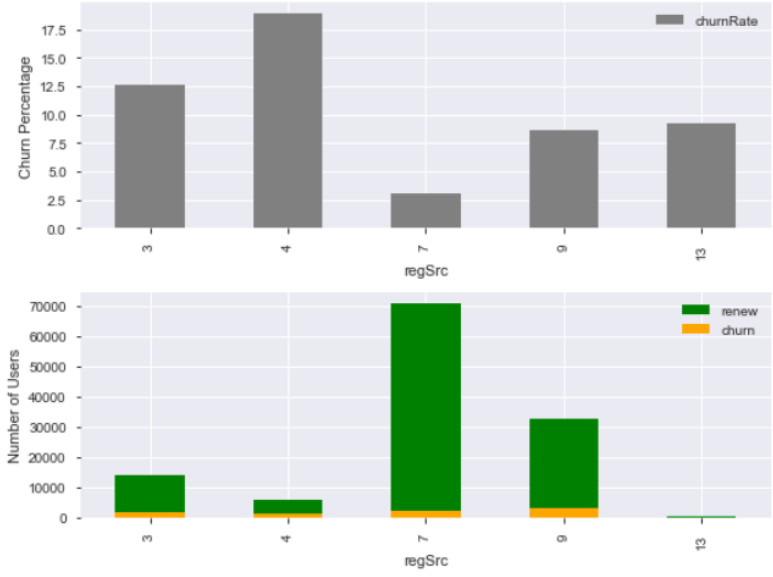
3. **Days since last transaction**: The feature has been derived to represent number of days since user has had any sales transaction. From the below chart, it can be inferred that if a user had last sales transaction beyond 29 days, he is likely to churn, though number of users who have had last transaction beyond 29 days harbors only 9% of total users.



User Registration

Let's change gears to dig deeper into the registration/membership data.

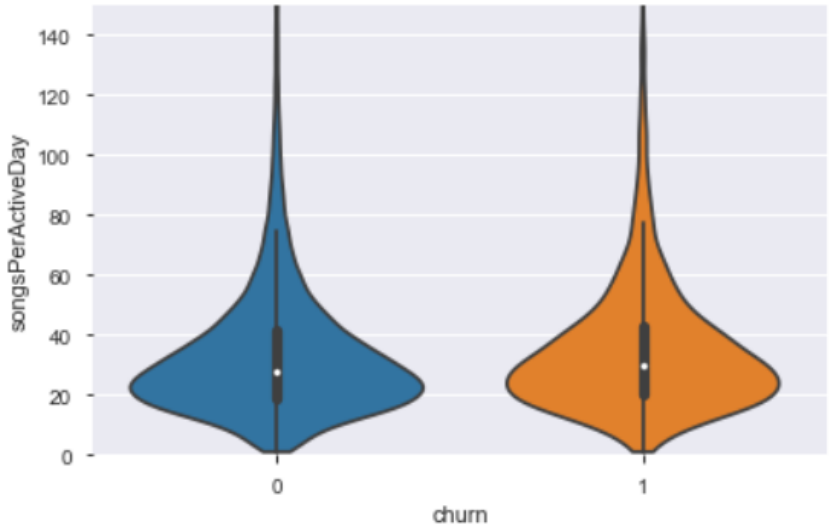
4. **Registration source**: Like payment method (from sales transaction data), registration method/source is a categorical variable, storing just five values – all integers. Analyzing the attribute shows that some registration sources/methods have higher churn rate than others.



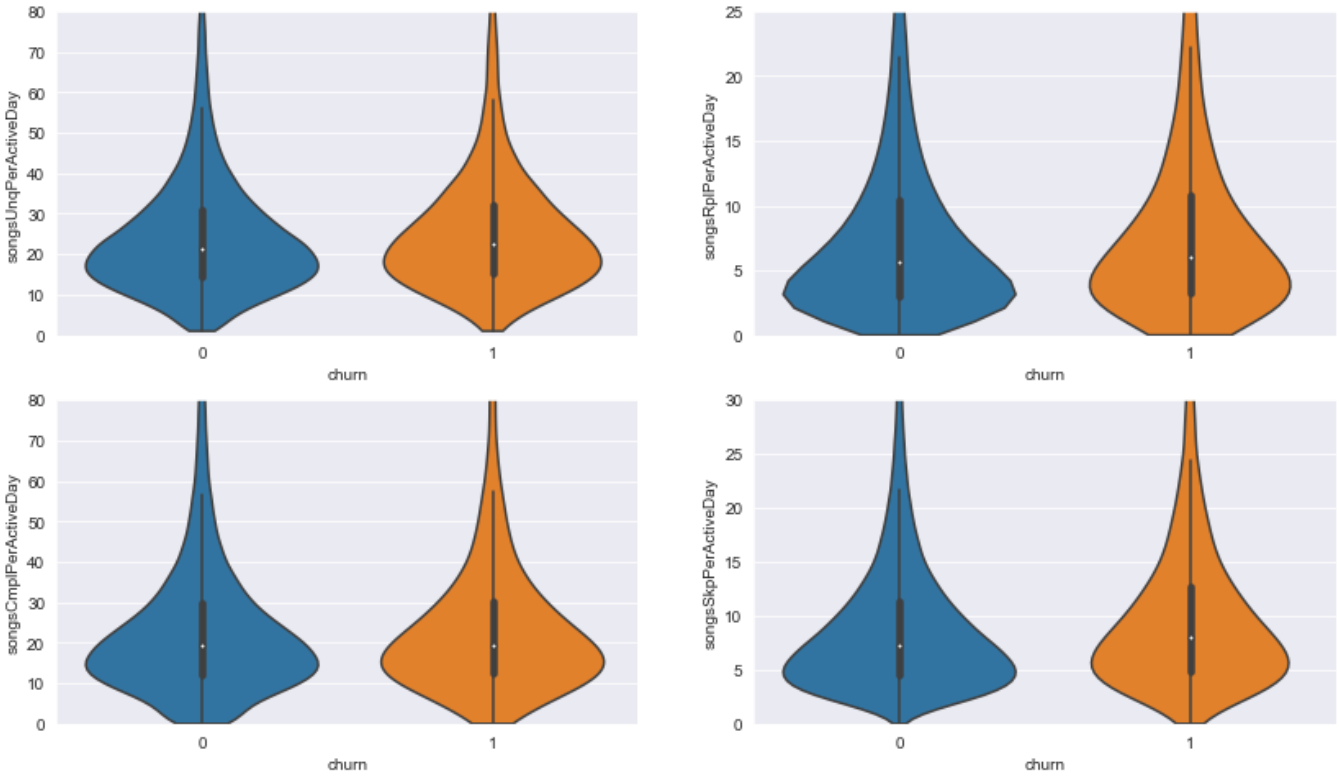
Lifetime Usage

Let's dive into usage data to see if renew users behave differently from churn users **during their lifespan**. Lifespan/lifetime of a user has been assumed to start from the day he used the service for the first time, until the current day, irrespective of whether the first usage day was much later than the day user registered or started paying for the service, and irrespective of whether the last login day was much earlier than the current day.

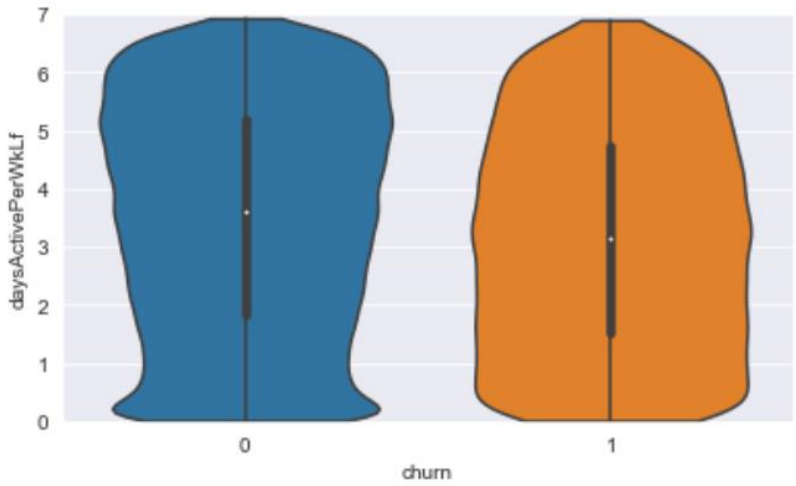
5. **Songs per active day**: Does a renew user, on average, listen to more songs on an active day than a churn user? The feature captures an *average* number of songs user listens to on a day he logs into the service. (It is derived as sum of all the skipped and completed songs in his lifetime divided by the number of days user logged into the service and used it, excluding the days he did not log into the application at all.) From the following distribution, one can conclude that, on an active day, both renew and churn users listen to same number of songs, on an average.



Similarly, there seems to be no difference between lifetime usage behavior of renew and churn users with respect to average number of songs played-once, replayed, completed or skipped on an active day.

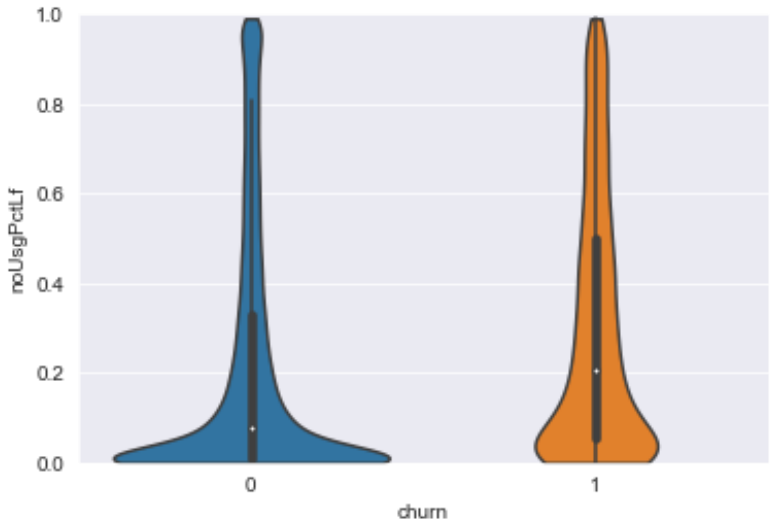


6. **Days active per week**: Does a renew user, on average, use the service more often during his lifetime than a churn user? The feature has been engineered to represent user's activeness on the service during his lifespan, by averaging number of days over all the weeks – since the week he started using the service till current date. The following distribution of renew and churn users shows that churn users *might* be recording slightly lower activeness over their lifetime than renew users.

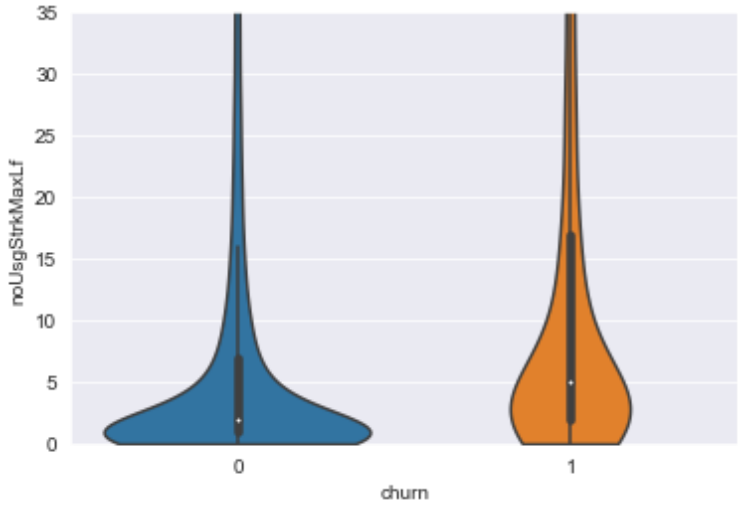


7. **Unused proportion of lifetime weeks**: Do churn users remain *inactive* for greater proportion of weeks available at their disposal, than renew users? The following distribution clearly states that

renew users logged into the application for higher proportion of weeks available during their lifetime, than churn users.



8. **Maximum no-usage streak (in weeks)**: Do churn users stay away from the service for longer periods (weeks in succession), on average, than renew users? The attribute has been devised to capture *longest successive* number of weeks user did not use the service for, during his lifetime. The following distribution shows that the maximum streak of churn users is more than that of renew users.

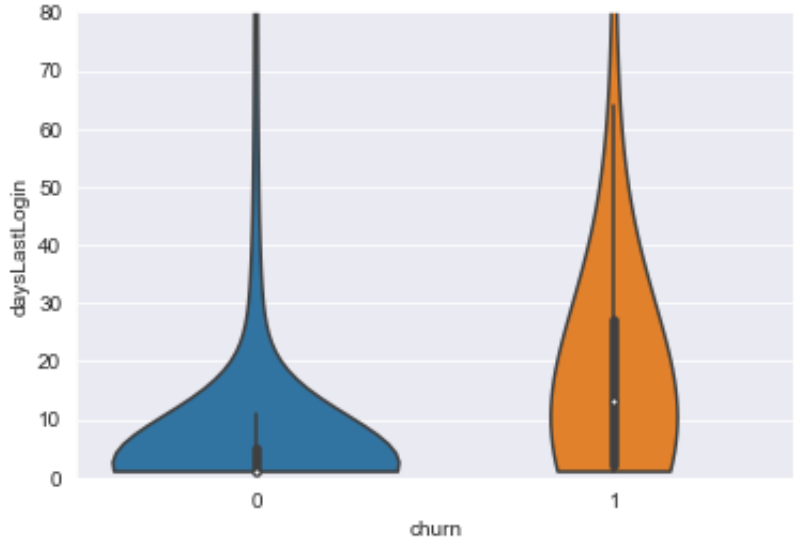


Recent Usage

Usage data is the most fascinating of all, in that we can wrangle the data in various ways and engineer new features/attributes to answer real-life questions about user's behavior. Instead of focusing on just lifetime usage, it might be worth looking into **recent behavior** of users, and more interestingly, **relative to longer usage time periods**.

Imagine you have a Netflix, Spotify or Amazon Prime (Video/Music) subscription account. Are you planning to stop the subscription to any service? If yes, do you see a relation between your departure decision and your *recent* usage behavior? Have you already stopped using the service completely? If not, has your usage declined these days, compared to what it used to be *earlier*? Let's create some useful metrics, and gain additional, and perhaps, more valuable insights than we have had so far.

9. **Days since last login**: Did churn users *last* log in and use the service lot further ago than renew users? The attribute has been derived to capture number of days a user has been continuously disengaged for since *last time* he logged into the service and used it. As expected, the following distribution indicates that the churn users have not logged into the service until the present day for longer time than renew users. The median number of days since last login for churn and renew users is 13 and 1 respectively.



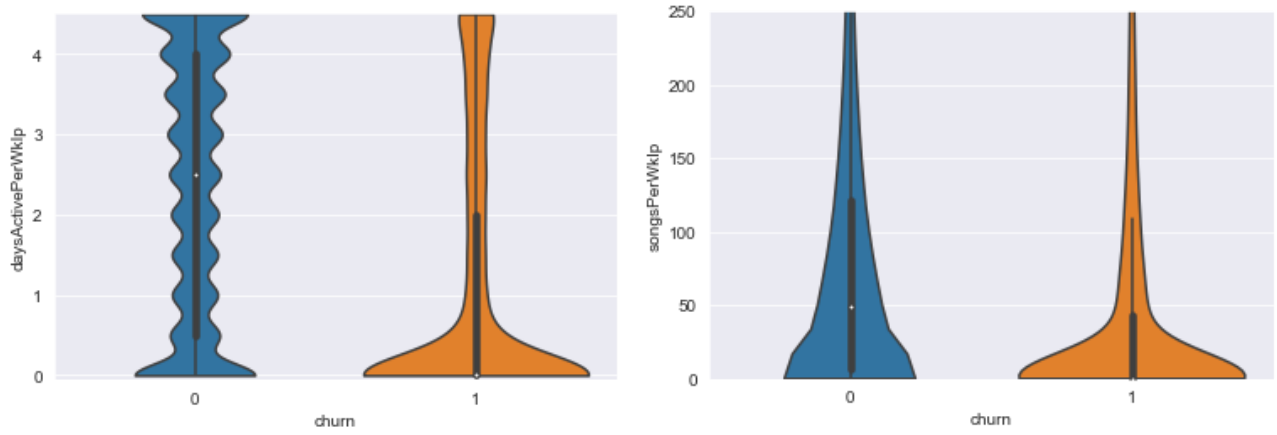
Following three metrics have been engineered to represent usage behavior for every user.

a) **daysActivePerWk –** Average number of days per week user was active on the application.
b) **noUsgWks –** Total number of weeks user did not use the service at all.
c) **songsPerWk –** Average number of songs per week user played; includes both played once and replayed.

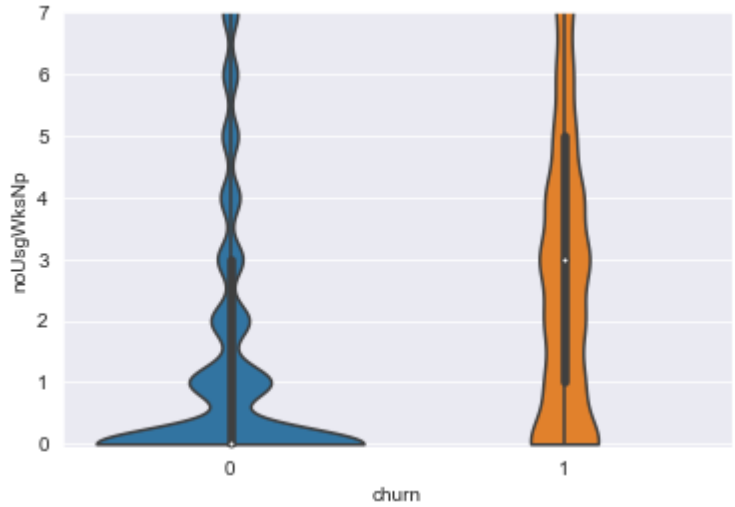
Following three time-periods have been formulated to unearth behavioral changes over time.

a) **Immediate-past**, abbreviated as "Ip", denotes usage behavior in last *two* weeks.
b) **Near-past**, abbreviated as "Np", denotes usage behavior in last *eight* weeks.
c) **Lifetime**, abbreviated as "Lf", denotes usage behavior since the date user started using the service for the first time until today (end of Feb' 2017).

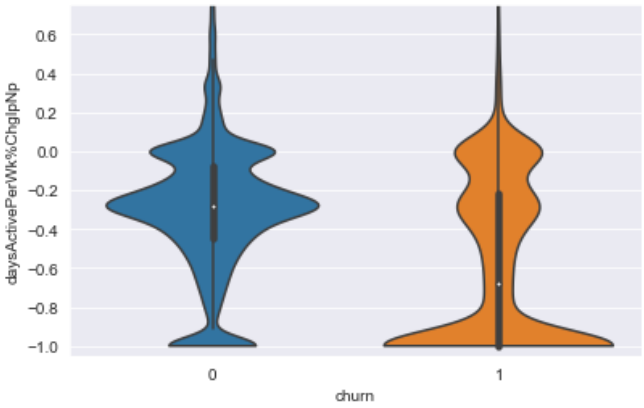
10. **Immediate-past behavior:** In the last two weeks, churn users have been less active on the service and have listened to less songs per week than renew users.

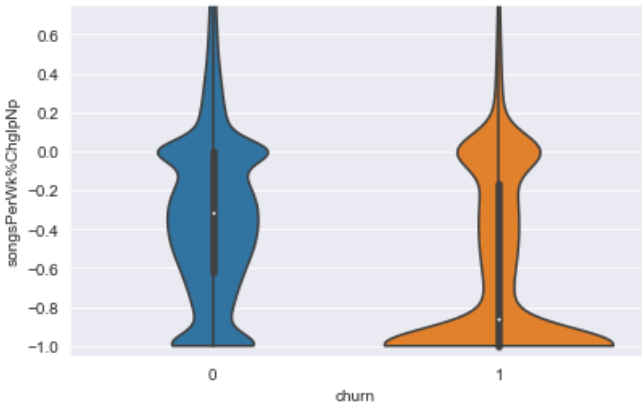


11. **Near-past behavior**: In the last eight weeks, churn users haven't used the service for more weeks (not necessarily in continuation) than renew users.

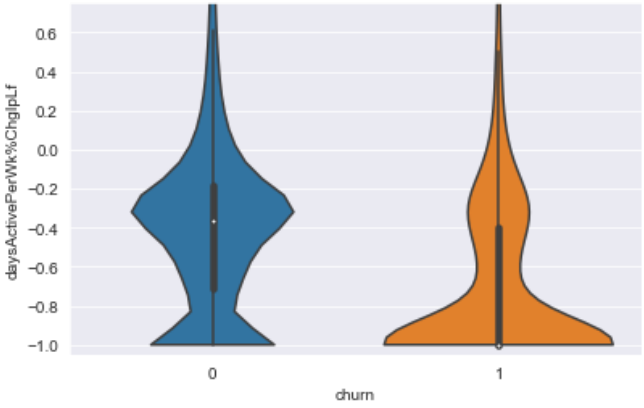


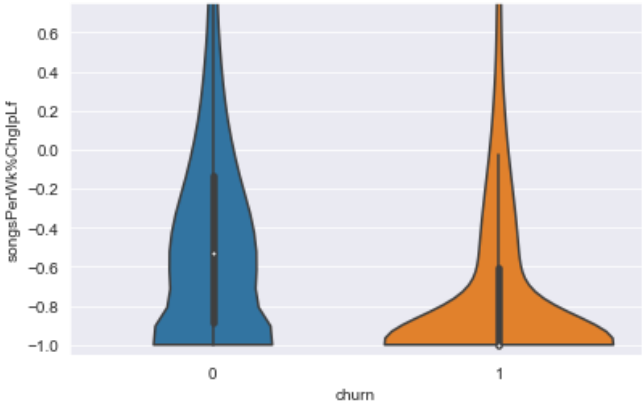
12. **Immediate-past behavior vs. near-past/lifetime behavior:** Comparing usage metrics, such as days active per week and songs played per week, for immediate-past with those for near-past yields some interesting facts. Compared to near-past usage, usage of churn users in immediate-past has declined by a higher degree than that of renew users.





Similarly, when compared to lifetime usage, usage in immediate-past of churn users has declined by a higher degree than that of renew users.





After exploring features representing recent usage behavior and change in usage patterns over time (immediate-past vs. near-past/lifetime), there was a dire need to see if such features would be useful in model building. As a result, model building had to be limited to just the users who started using the streaming service at least eight weeks back, especially so when the volume of users who started using the service in last eight weeks is not significant compared to total number of sample users available (refer to **Data Extraction** under Data Wrangling section).

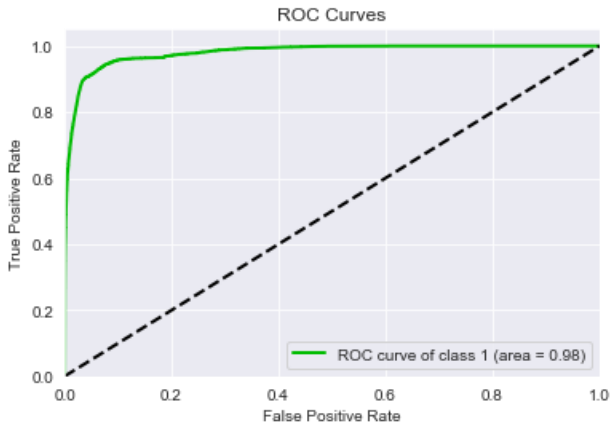
Predictive Modeling

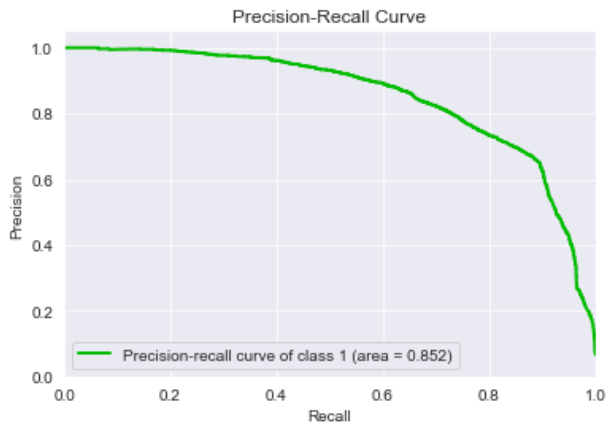
Post-EDA, following 26 features were selected in both training and holdout sets, and then normalized within a range of 0 and 1, so that no feature could bias the model in favor of users who had abnormally large values due to long association to and use of the service.

The training set was used to fit Extreme Gradient Boosting (XGB), Random Forest and Bagging models and tune their hyperparameters using Stratified 5-Fold cross validation (to maintain original share of churn users as ~6.5% of total users in all the folds). The following results of testing the models on unseen data (hold-out set) clearly states that each model outperformed the naïve classification model having base log-loss score of 0.24 on the holdout set. Since XGB outperformed both Random Forest and Bagging in every aspect, it has been selected as the final classification model for prediction purposes.

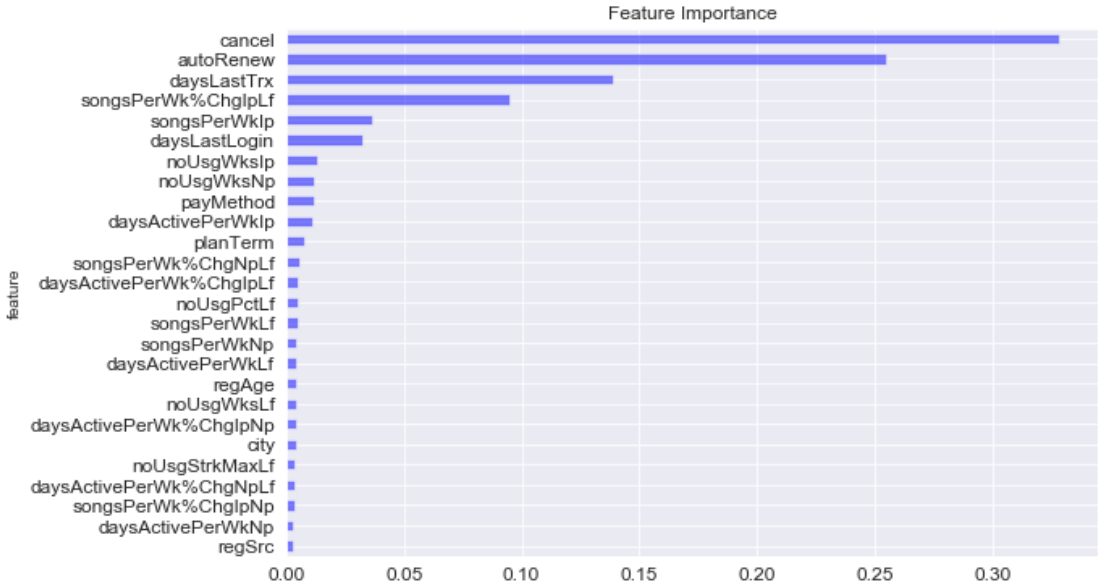
Base Log-Loss	0.241		
	XG Boosting	**Random Forest**	**Bagging**
Log-Loss	0.0752	0.0806	0.1014
Precision	83.1%	83.4%	83.5%
Sensitivity	68.7%	67.8%	64.8%
ROC AUC	0.98	0.976	0.972

ROC Curve of the XGB model shows the area under the curve as ~0.98, which is much better than baseline score of 0.5 (denoted by dashed black line below) and indicative of a good and robust classifier.





Using feature importance capability of XGB implementation in scikit-learn, all the input features were ranked based on their importance in constructing boosted decision trees. The more an attribute is used to make key decisions in decision trees, the higher its relative importance.



Upon highlighting top 10 features (in green cells below) based on their importance score, it is evident that features that have been engineered out of last sales transaction and recent usage have substantial effect on the predictive model. All other features related to registration and lifetime usage rank among the lower half of the set.

S. No.	Data Category	Feature	Importance	Rank
1	Last Sales Transaction	cancel	0.3286	1
2		autoRenew	0.2547	2
3		payMethod	0.0115	9
4		planTerm	0.0076	11
5		daysLastTrx	0.1384	3
6	User Registration	regAge	0.0039	18
7		city	0.0037	21
8		regSrc	0.0025	26
9	Lifetime Usage	noUsgWksLf	0.0039	19
10		noUsgPctLf	0.0048	14
11		noUsgStrkMaxLf	0.0037	22
12		daysActivePerWkLf	0.0040	17
13		songsPerWkLf	0.0047	15
14	Recent Usage	daysLastLogin	0.0325	6
15		noUsgWksIp	0.0128	7
16		daysActivePerWkIp	0.0110	10
17		songsPerWkIp	0.0367	5
18		noUsgWksNp	0.0115	8
19		daysActivePerWkNp	0.0029	25
20		songsPerWkNp	0.0041	16
21	Recent vs. Longer-term Usage	daysActivePerWk%ChgIpNp	0.0038	20
22		songsPerWk%ChgIpNp	0.0033	24
23		daysActivePerWk%ChgIpLf	0.0049	13
24		songsPerWk%ChgIpLf	0.0950	4
25		daysActivePerWk%ChgNpLf	0.0035	23
26		songsPerWk%ChgNpLf	0.0058	12

Conclusion

Based on the available datasets and some twisted feature engineering, 26 odd features for 85% of available users were leveraged to train and tune three tree-based models – XGB, Random Forest and Bagging. After testing the models on unseen data, XGB won the coveted spot for predicting user churn for KKBOX. The model can run as-is for users who have been using the service for at least eight weeks.

Based on the modeling results, a couple of things stand out that KKBOX needs to be cautious about.

1) It's paramount that a user stays on an auto-renewal plan, as the probability of continuing with the service on auto-renewal plan is very high. If the user cancels an auto-renewal plan, then he is likely to leave.
2) It's important that a user logs into the application and uses it every now and then. Couple of unused weeks in succession poses risk of churning.

As I conclude this report, I just realized that my Netflix usage has declined substantially in last two months, and that I have been away from the application for last three successive weeks. Now, I can't help but ask myself – Do I really need the service anymore? Should I continue with the subscription? Perhaps, Netflix's Data Scientists are tapping into its usage data and flagging me as a "risk-to-churn" user. Or perhaps, it's already done, and the next phase is in progress – a promotional email, a survey, or better yet (for me), a one-off discount!

Codebase

All the analysis has been conducted in Jupyter notebook after installing standard Anaconda distribution for Python 3. The entire code has been organized into three python notebooks and uploaded on Github repository.

1. *Data Wrangling.ipynb* comprises of sample data extraction, data quality assessment, data cleaning, data manipulation and feature engineering (which uses feedback from Exploratory Data Analysis).
2. *Exploratory Data Analysis.ipynb* explores each of the available datasets from various angles and discovers insights into potential feature set for model training.
3. *Model Building.ipynb* deals with training, tuning and evaluation of multiple statistical models.

Future Scope

No matter how well the prediction results turn out to be, time limitation on any project opens room for improvement of the predictive results in future iterations. This project is no exception to that belief.

Though the reader of this report might have different suggestions for future iterations, here are my two cents (or may be five, in this case) based on my tryst with this undertaking.

a) Use Spark and AWS to process the entire dataset available (for 1mn users) and not limit the sample to 150k users. Though I doubt that it would improve the prediction results substantially, if any, it would certainly bring the model closer to real-life production deployment, since an ideal requirement would be to run the model for *all the users* every month.

b) Even though the current model provides prediction for roughly 85% of available users, it might not be a bad idea to build separate model(s) to predict for users who recently started using the service (refer to **Data Extraction** under Data Wrangling section) and/or for users for whom usage and registration information is not available (refer to **Source Data** under Introduction section). For such users, we could use just the features related to last sales transaction for model building and testing.

c) It would surely be interesting to explore Support Vector Machines (SVM) for this classification problem and see if it yields better results than XGB, because unlike all the algorithms explored so far, SVM is not based on decision trees, and could perhaps surprise us. I skipped it from consideration because one run of the model fitting with default hyperparameters took more than half-an-hour; it seemed prohibitively expensive to train SVM with 128k records. Perhaps, converting datatypes of few attributes to *category* in pandas might speed things up.

d) If KKBOX supplied information about user's profile such as number of playlists stored, number of songs favorited or saved in playlists, etc., prediction capability could be further improved. Such attributes could likely turn out to be good predictors of user's decision to churn, as they are important factors of customer's stickiness to such a service. I, myself, am hooked to Spotify because of the sheer number of songs searched and saved in my playlists. Who has the time to use another platform and manually search for hundreds of songs and create playlists all over again?

e) KKBOX should try to put controls on data quality during data input from users. User Age could turn out to be a useful predictor only if we could rely on its accuracy. It's unreasonable to expect that users would provide *correct* private information, such as birth date and/or age, during registration or membership enrollment. KKBOX's marketing division could instead ask users of range their age falls into (such as 18-25, 26-32, etc.), which is more likely to be supplied *correctly* by the users, at the expense of *precision*.