Sparkify Capstone Project Documenation

Introduction

Sparkify is a fictious music streaming app, which contains two months of user behavior log which is around 128MB in size. This dataset emulates the data similar to that of a regular music streaming app (Imagine something like Spotify). The user log contains some basic information about the user as well as information about every single action that is captured on the app during every individual session. A user can have several entries. Major problem with the Sparkify dataset, just like every other business which revolves around consumer is, Customer Churn. Sparkify is no exception as it also deals with customer churn and how do we deal with addressing such a fictious problem would be relevant to the real world problem of Customer Churn.

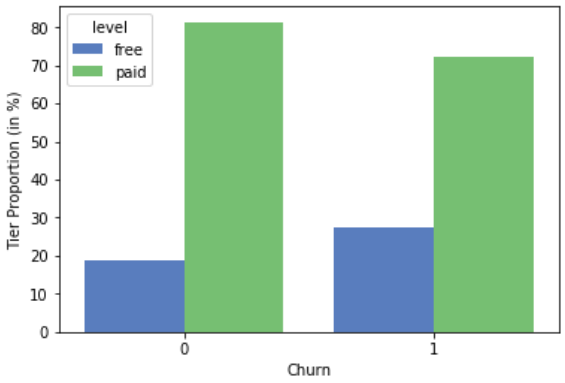
What is Customer Churn and How do we predict which customers are likely to Churn?

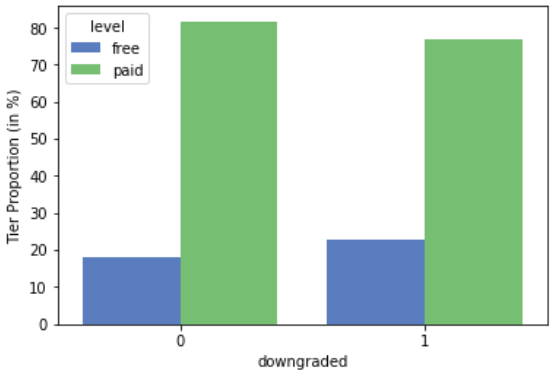
Customer Churn in general means, when an existing customer stops doing business or ends the relationship with a company. In the Sparkify context, a part of the user is churned and this can be identified through action such as cancellation of their account. In some cases, the user might have downgraded their subscription from paid to free user. Such actions are eventually considered as churn and they can be distinguished in a certain time frame, complemented with other actions. We will work on the given data set and explore them to find the relevant attributes which will help us predicting churn.

Data Exploration

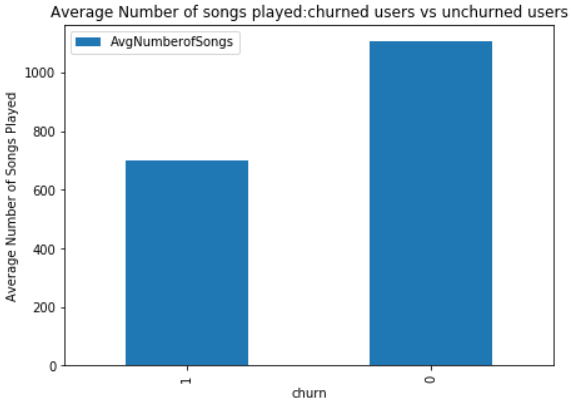
There are 18 features in the dataset such as : `Artist`, `auth`, `firstName`, `gender`, `itemInSession`, `lastName`, `length`, `level`, `location`, `method`, `page`, `registration`, `sessionId`, `song`, `status`, `ts`, `userAgent`, `userId`. However, one feature namely page forms the basis for the rest of the analysis. Page feature has several events/activity triggered by each users. These activities helps us to understand how the user gets Churned.

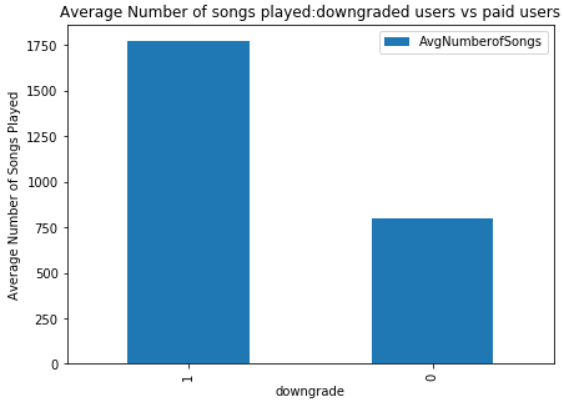
We can define a churn, when a “cancellation confirmation” event is triggered by a user. Likewise, we can define downgrade, when a user triggers “submit downgrade”. When we group both Churn & downgrade by level namely: free & paid, we get the below distribution. Plot on the left suggests that 80% of the users who have not churned are paid user while the 20% are free users. Contrarily on the churned population, 70% were paid and 30% were free user. Plot on the rights reveals us that nearly 76% of the users who downgraded were paid users and the remaining 24% were free users. The users who have not downgraded their number are in proportion with users who have not churned.



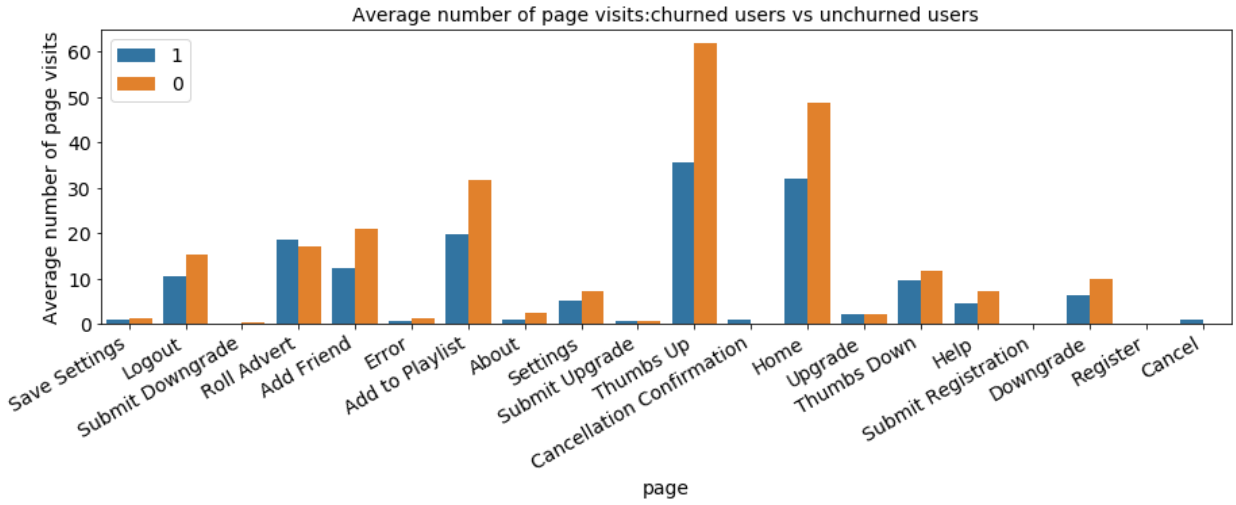


On an average, number of songs played by churned user is around 700 songs. When we look at the downgraded subset, it tells us that the users who downgraded on an average listen to 1750 songs and in comparison with users who have not churned only listens to 1100 songs. May be this is already an indication that it appears that users who downgrade are actually heavy users who listen to about 30-50% more as many songs on average than the average paid user. A possible reason could be that heavy users could be drawn to another type of service; as a result, they are downgrading.

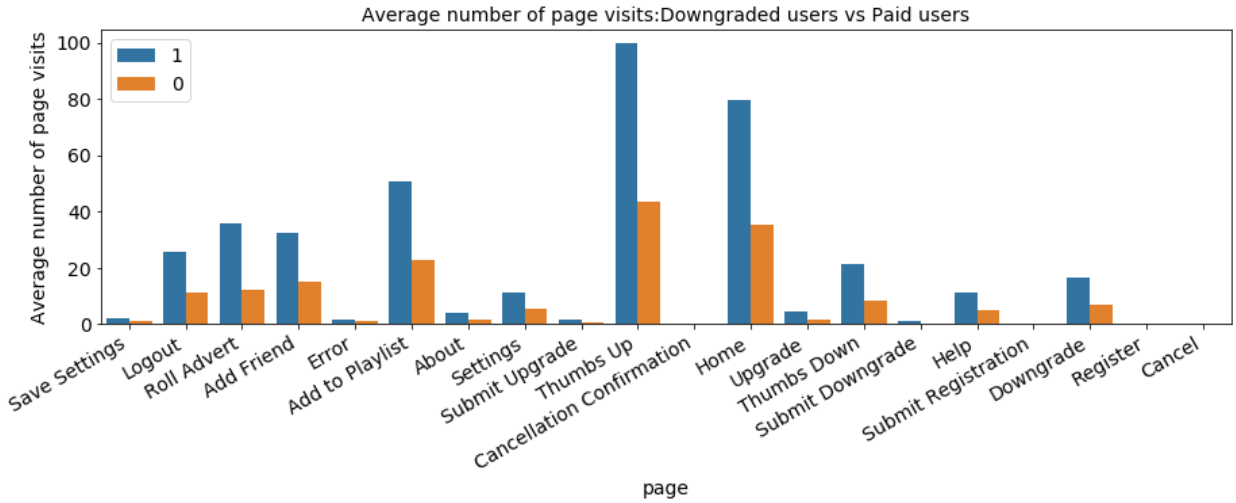




In the below bar plot, in x-axis we look at the average number of page visits in comparison with each page events on the y-axis for every churned user Vs active user. Certain action such as roll advert, logout, settings, Thumbs down & downgrade are proportionally dominant behaviors spotted on the churned users.



Let us compare the average number of page visit with respect to the page events on the downgraded vs Paid users. It becomes clearer that roll advert, logout, thumbs down and downgrade are spotted higher in proportion on the downgraded user than the paid users.



Preprocessing & Feature Extraction

It was important to understand how the user's activity change over time. Being an active user immediately after sign-up and then becoming a dormant user around that time the user churn. So it is important to look at user activity within certain time bins. Some might wonder - why is it essential to look at each activity in time bins? Answer to this question is simple, since subscriptions are usually monthly based and in some cases annual based. Considering the given dataset, we only have 2 months of data. Hence it makes sense to look at the activity in weekly windows and try to

make predictions for the current window and for the next window. If the 1-week window does not fit, then switch it to 2 weeks' time frame and continue the below analysis. These 1-week time windows will also help to normalize/smooth out any noise or irregularities on the day to day usage of the service. These time bins are built from the perspective of a user, so that time bin 0 is the first 7 days of activity for a user's account and bin 1 would be next 7 day and so and so forth — this normalizes the time bin number meaning. Within user time bins, I have extracted some new features:

1. PaidInTimeBin and FreeInTimeBin — These are binary features about the user's subscription level, in a particular time bin which determines if they change their subscription level or not.
2. ChurnedInTimeBin, DowngradedInTimeBin, UpgradedInTimeBin, PreviouslyDowngraded, WillChurnInNextBin, and WillChurnSoon — These are binary features about user changes to their account and these can be used as features and also as labels for our models to predict churn.
3. OneHotEncoding of the page name with the count of page visits from the user in the time bin. Note: I throw out some of the pages that we already have encoded (e.g., 'Cancellation Confirmation' == ChurnedInTimeBin)
4. SessionsInTimeBin, ArtistsInTimeBin, and DistinctSongsInTimeBin — these are count distinct features of sessions, artists, songs for the user within the time bin
5. UserId with Null records were excluded as it implies no meaning and all of the engineered features were converted to integers with nulls replaced as zeros.

Modeling

Since the focus is on predicting churn, I decided to use the below features as labels: ChurnedInTimeBin, DowngradedInTimeBin, UpgradedInTimeBin, PreviouslyDowngraded, WillChurnInNextBin, and WillChurnSoon and applied 3 different models (Logistic Regression, Random Forest and GBM) to see which one does a closer prediction. This resulted in 18 models (6 X3) in total. In which 3 out of the 6 labels namely: ChurnedInTimeBin, WillChurnInNextBin, and WillChurnSoon yielded decent results.

Considering the 2 weeks' time bin, distribution of value across the aforementioned label suggests that positive churn label occurs only in 7.67% of the dataset in average resulting in a class-imbalance problem.

	1	0	1	0
ChurnedInTimeBin	52	809	6%	94%
WillChurnInNextBin	39	822	5%	95%
WillChurnSoon	91	770	12%	88%

Model Evaluation

Having already expressed that the given dataset has some serious class-imbalance problem, the best evaluation criteria would be to use Area Under the Curve Precision Recall (AUC PR). Morever, the label WillChurnSoon was significantly the best label to predict the churn while Logistic Regression was the better classifier as it accurately predicted the churn. Although the average AUC PR of Logistic Regression in comparison with GBM was close enough but Logistic Regression out performed to be a winner in terms of run time. Average runtime for Logistic Regression was 20 mins while that of GBM was 80 mins which is 4 times longer.

AUC PR for Model/-Label

	Logistic Regr.	Random Forest	GBM	Label Average
ChurnedInTimeBin	0.6349	0.4417	0.6211	0.5659
WillChurnInNextBin	0.8426	0.4864	0.8341	0.7210
WillChurnSoon	0.7678	0.7456	0.7540	0.7558
Classifier Average	0.7484	0.5579	0.7364	

Conclusion & Future Work

Multiple binary models helped us to make more accurate prediction and also gave a better control. However, class-imbalance problem poses to be threat which could be potentially dealt with some upsampling or downsampling of the dataset, in addition we could also try using SMOTE for class imbalance problem. In addition to exploring the churn and downgrade activity in time bin, we could also extend this to page activity in different time bin and use them as a feature alongside the existing labels and combine them into a Multi-class Multi-label prediction problem.

References

1. The relationship between Precision-Recall and ROC curves - https://dl.acm.org/citation.cfm?id=1143874
2. https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/
3. https://docs.databricks.com/spark/latest/mllib/binary-classification-mllib-pipelines.html
4. https://towardsdatascience.com/machine-learning-with-pyspark-and-mllib-solving-a-binary-classification-problem-96396065d2aa