**Data Science Project Protocol**

*Author(s):*

*Adi* *Vaknin*

# Introduction

Here you have to give some known facts about the field you will work on.

Try to focus on the problems that are most common and then state the goals of the project.

answer the following question:

* Which questions do we want to answer?

In this challenge, we were asked to predict **whether a user will churn after his/her subscription expires**. Specifically, we want to forecast if a user makes a new service subscription transaction within 30 days after the current membership expiration date.

* What is known about the problem?

KKBOX offers subscription-based music streaming service. When users sign up for the service, users can choose to either manual renew or auto-renew the service. **Users can actively cancel their membership at any time.**

The **churn/renewal definition can be tricky** due to KKBox's subscription model. Since the majority of KKBox's subscription length is 30 days, a lot of users re-subscribe every month.

* How do we define the outcome(s)?

‘**is\_churn**’ is the target variable. Churn is defined as whether the user did not continue the subscription within 30 days of expiration. is\_churn = 1 means churn,is\_churn = 0 means renewal.

* What is known to influence the outcome?

The key fields to determine churn/renewal are transaction date, membership expiration date, and is\_cancel. Note that the is\_cancel field indicates whether a user actively cancels a subscription. Subscription cancellation does not imply the user has churned. A user may cancel service subscription due to change of service plans or other reasons. The criteria of "churn" is no new valid service subscription within 30 days after the current membership expires.

* Do we have any possible new knowledge that has not been in use before?

Based on the information we found on the internet the most common music streaming groups age are: 16-23,25-34,35-44,45-54,55-64,65+. This information helped us to divide the ‘bd’ (age) column to age group columns and turn outliers to 0.

This part must be between half to one and half page.

# Methodology (Project design)

## Data

Describe how you plan to manipulate the data. Please answer the following questions:

* Which data will be used?
  + Describe data sources.

KKBox’s Churn Prediction Challenge which is hosted on Kaggle: <https://www.kaggle.com/c/kkbox-churn-p>

* + Describe possible external data sources that may enrich our data.
  + Data for external validation?
* On which time frames periods your project will be based on?
  + Timeframe for training?

KKBox has provided a lot of data, typically in terms of 2 versions. The version 1 contains the data till February 2017. In between the competition period they also added version 2 data, which contains the data till March 2017.

* + Timeframe for test?

The test data is with users whose subscription expires within the month of March 2017 and April 2017.

* + How do you define your subjects?
  + Inclusion criteria?
  + Exclusion criteria?
* Which would be your outcome variable?
  + ‘is\_churn’ means customer does not resubscribe within 30 days of current membership expiry.
* Are there confounder variables that may affect the outcome?
* Is there a possible source of bias in our data?

In general, it is expected that the number of churned customers will be smaller than the number of customers who have not churned. Most of the time there are only a few cases for the churned class, and the majority will be for the non-churned class. This makes it difficult to build unbiased models.

* Describe your data exploration strategy.

we do an exploratory data analysis to understand the available data and decide on necessary preprocessing. Exploratory data analysis will enrich the feature engineering process too. This involves finding the value range, distribution, missing values, correlation with each other and target variable (i.e., churned or not), and plotting data to understand trends.

The next step is to clean up the data. This involves fixing missing or wrong data, removing outliers, fixing cardinality of categorical features, and removing any features that are not useful or are biased.

* Which techniques will be applied to enrich the data?

Feature engineering plays a major role in churn prediction because, in order to be robust, the model should be fed with information that exhibits the churning/non-churning behavior. For example, “discount\_price” can be considered as the difference between “plan\_list\_price” to “actual\_amount\_paid”. Features captured from different time frames add value to the model such as “membership\_duration” is the difference of “membership\_expire\_date” to “transaction\_date” or “amt\_by\_day” is the ratio between “actual\_amount\_paid” to “payment\_plan\_days”.

We used with another feature engineering technic “Bin-counting”. This technic suited to categorical variables with large number of categories. For instance, in column city there are 21 different cities. Based on the number of every city frequency appearance and the total number of all cities count frequency, we can calculate the “city\_frequency\_prcnt”.

Feature extraction also help us enrich the data. From dates columns we obtained day month and year.

In addition, we used in “Feature Transformation” technics such as One-hot-encoding to transform categorical data such as “gender” with female and male categories to binary columns with 0 (false) or 1 (true) of “is\_female” or “is\_male” (only one column is enough).

* How will you deal with outliers?

Box plots or histograms to understand how the variables distributed and Interquartile Range method are used to detect outliers. Depending on the case, these numeric outliers are transformed by applying mathematical functions such as log or sqrt in order to build a normalized dataset.

* How you will deal with missing values

For categorical variables we will add indicators columns called is\_na to indicate which cells are empty (in conjunction with One-hot-encoding columns.

For numeric variables we will use technic of handling missing data through MICE or Multiple Imputation by Chained Equation. It is a sophisticated approach is to use the Iterative Imputer class, which models each feature with missing values as a function of other features and uses that estimate for imputation.

* Add at the end of the protocol (appendix) the [Data retrieval protocol](https://docs.google.com/spreadsheets/d/1pYYjgwZ_8PS1Bcmc2kRNHTL0f_rk__GCJALLs1JHPUQ/edit#gid=0)

## Models

Describe how do you plan to develop your models:

* How do you plan to divide your data?
  + Training, validation, test - proportions, techniques

Since it is a Kaggle challenge the data is already split to test and train.

Using in pyMechkar train\_test function which generates a training and test dataset and checks if it is well balanced based on p-value < 0.05. Set the proportion to 0.7, indicating that the training dataset will contain 70% of the cases and the validation dataset will contain the 30% of the cases.

The pyMechkar train\_test function also uses in Sklearn train\_test\_split function. The train\_test\_split function Split arrays or matrices into **random** train and test subsets.

* Do you need to balance your data? How?

With a churn rate of approximately 12%, there is a moderate class imbalance issue.

We defined imbalanced data as smaller/equal to threshold of 10%. Since 12% is above threshold, we will not use over-sampling technic but on train a model with class weights in the finetuning model stage.

The goal of class weight is to identify fraudulent transactions, but you don't have very many of those positive samples to work with, so you would want to have the classifier heavily weight the few examples that are available. You can do this by passing weights for each class through a parameter. These will cause the model to "pay more attention" to examples from an under-represented class.

* Do you need to stratify/subsample your data? How?

We used in subsample parameter in the final Xgboost stage the subsample denotes the fraction of observations to be randomly samples for each tree.

* What techniques will you apply to model your outcome?
  + Unsupervised
  + Regression
  + Classification

We should pick a few machine learning algorithms. We will choose between tree-based (e.g., XGBoost, Random Forest) and numerical (Deep Neural Networks, Recurrent Neural Networks) models. Picking the right model depends on many factors including one’s expertise in machine learning/artificial intelligence, the size of the data set, and the nature of the data set.

I have experimented various machine learning models:

* + - Logistic regression
    - Decision tree
    - Random forest
    - KNN
    - AdaBoost
    - XGBoost
* Will you use cross-validation and/or bootstrap?

No.

* Which measures you will use to train and evaluate your models? Why?

Models were evaluated using the AUC (Area Under the ROC Curve). Accuracy is misleading when classes are imbalanced. A method of always classifying a member as “not churning” would lead you to 80% accuracy.

* Do you plan to use ensembling or will use your best model?

We will use in the best model.

## Deployment of your model

* Who will make the QA of the project?
  + Which units will be assessed?

An independent testing team may need to be supplemented by third-party experts or the QA team in the KKBOX company, but they must have expertise in modelling, advanced analytics algorithms, numerical computing, commercial and open source packages for analytics and data science, and deployment of systems embedding advanced analytics.

* + Write a QA protocol for each step of the project.
    - interview stakeholders from business and analytics development to understand the business problem and context.
    - review existing models and procedures.
    - review data sources
    - implement models in alternative technologies to compare results — languages, solvers, analytics engines.
    - experiment with models and a variety of test data sets to uncover issues and stress the model implementation.
    - suggest improvements and recommend possible further investigation.
* Who is the final user of the predictions?

The music streaming service providers KKBox. A high customer churn rate will hit KKBox finances hard. Marketing department are also may be the final user - the marketer has to know exactly what marketing action to run on each individual customer to maximize the chances that the customer will remain a customer.

* How will the prediction be presented to the final user?

As a dynamic dashboard.

* How will the final user be trained to use and interpret the prediction?

For quick-churn customers, what’s vital is to reinforce the value of your offer. Since they were once interested in your business, it would be easy enough to spark up that interest again. Revisiting the points that made them leave the service, reoffering the services, and having access to in-depth and user-friendly onboarding instructions and content will help them dive back into the company’s services.

* On which platform the predictions will be deployed?

Another common method in using a machine learning model in production is to Integrate the model into existing systems within an organization. This has the benefit of high data security since data will not move outside the network of the organization. In-house servers can be used to host the model and existing data warehouses can be used to fetch data.

* + How frequently will the model be updated?

To stop the model from deteriorating, **continuous updating** of customer databases and model tuning is needed.

* What will happen in cases where the model returns a null prediction (eg. incomplete data)?

As mentioned before, on the same way we handled missing values, we will use technic of handling missing data through MICE or Multiple Imputation by Chained Equation. It is a sophisticated approach is to use the Iterative Imputer class, which models each feature with missing values as a function of other features and uses that estimate for imputation.

* Which measurements will be used to evaluate if the prediction is decaying?

Concept drift driven model decay can be expensive to fix given the need for manual re-labeling of old data. Drift can be detected via global measures such as worsening f1-scores for sampled new data. We could do a complete re-label of all data here as our labeling is not manual but rather formula driven. It will likely not be possible in a real situation, however. Even a rough identification of the old data that needs to be re-labeled could help with lowering the manual work. Something that requires further work.

* How will the model’s prediction be adapted for the needs of the production system?

Further parameter optimization of the XGBoost and further exploration of additional feature engineering not yet tested.

# Results

Here you will present the main results of all the processes. We will describe:

* The final amount of data used (total, train, test, etc)

full\_train\_final\_amount: 1,082,190

full\_test\_final\_amount: 1,076,941

* The number of outliers and the way of treating them.

With log and sqrt transformation as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| column\_name | outliers\_before\_log | outliers\_after\_log | percentage\_change |
| sum\_num\_25 | 66084 | 18734 | 0.716512 |
| sum\_num\_50 | 61068 | 21932 | 0.640859 |
| sum\_num\_75 | 56203 | 252 | 0.995516 |
| sum\_num\_985 | 62101 | 860 | 0.986152 |
| sum\_num\_100 | 58786 | 33819 | 0.42471 |
| sum\_num\_unq | 48514 | 28685 | 0.408727 |
| sum\_total\_minutes | 56216 | 32225 | 0.426765 |
| mean\_num\_25 | 56823 | 7056 | 0.875825 |
| mean\_num\_50 | 55561 | 19300 | 0.652634 |
| mean\_num\_75 | 46400 | 20378 | 0.560819 |
| mean\_num\_985 | 52243 | 23597 | 0.548322 |
| mean\_num\_100 | 60313 | 33401 | 0.446206 |
| mean\_num\_unq | 46480 | 23825 | 0.487414 |
| mean\_total\_minutes | 58906 | 30450 | 0.483075 |
| transactions\_count | 620 | 13878 | -21.383871 |
| payment\_method\_id\_mode | 98881 | 98881 | 0 |
| payment\_plan\_days\_mean | 165937 | 167320 | -0.008334 |
| plan\_list\_price\_mean | 60620 | 52047 | 0.141422 |
| actual\_amount\_paid\_mean | 61634 | 53569 | 0.130853 |
|  |  |  |  |
| column\_name | outliers\_before\_sqrt | outliers\_after\_sqrt | percentage\_change |
| discount\_price\_mean | 106023 | 72990 | 0.311564 |
| membership\_duration\_mean | 255037 | 253789 | 0.004893 |

* The number of missing values and the methods used for imputing them.
* For categorical values we imputed all missing values by simply replacing them with a 0 in the categorial column and created indicator na column with a 0 or 1 to indicate the new value 0 indeed created by the imputation process.
* For numeric columns we imputed the missing values with MICE imputation (based on knn).
* The amount of missing value as follows:

|  |  |
| --- | --- |
| Column | # of NAN |
| user\_logs\_count | 180013 |
| city | 120759 |
| bd | 641597 |
| registered\_via | 120759 |
| is\_female | 120759 |
| is\_male | 120759 |
| registration\_init\_time\_Year | 120759 |
| registration\_init\_time\_Month | 120759 |
| registration\_init\_time\_Day | 120759 |
| Age\_10-19, | 120759 |
| Age\_20-24 | 120759 |
| Age\_25-34 | 120759 |
| Age\_35-44 | 120759 |
| Age\_45-54 | 120759 |
| Age\_55-65 | 120759 |
| Age\_65+ | 120759 |
| log\_sum\_num\_25 | 180013 |
| log\_sum\_num\_50 | 180013 |
| log\_sum\_num\_75 | 180013 |
| log\_sum\_num\_985 | 180013 |
| log\_sum\_num\_100 | 180013 |
| log\_sum\_num\_unq | 180013 |
| log\_sum\_total\_minutes | 180013 |
| log\_mean\_num\_25 | 180013 |
| log\_mean\_num\_50 | 180013 |
| log\_mean\_num\_75 | 180013 |
| log\_mean\_num\_985 | 180013 |
| log\_mean\_num\_100 | 180013 |
| log\_mean\_num\_unq | 180013 |
| log\_mean\_total\_minutes | 180013 |
| sqrt\_discount\_price\_mean | 291996 |
| sqrt\_membership\_duration\_mean | 403 |

* The distribution of the data (timeframes)
* The methods used to transform the data and to generate new features.
* Feature engineering plays a major role in churn prediction because, in order to be robust, the model should be fed with information that exhibits the churning/non-churning behavior. For example, “discount\_price” can be considered as the difference between “plan\_list\_price” to “actual\_amount\_paid”. Features captured from different time frames add value to the model such as “membership\_duration” is the difference of “membership\_expire\_date” to “transaction\_date” or “amt\_by\_day” is the ratio between “actual\_amount\_paid” to “payment\_plan\_days”.
* We used with another feature engineering technic “Bin-counting”. This technic suited to categorical variables with large number of categories. For instance, in column city there are 21 different cities. Based on the number of every city frequency appearance and the total number of all cities count frequency, we can calculate the “city\_frequency\_prcnt”.
* Feature extraction also help us enrich the data. From dates columns we obtained day month and year.
* In addition, we used in “Feature Transformation” technics such as One-hot-encoding to transform categorical data such as “gender” with female and male categories to binary columns with 0 (false) or 1 (true) of “is\_female” or “is\_male” (only one column is enough).

# Conclusion

Here you will write about how the project began, which were the most important challenges you had when developing the project, and how did you get the final prediction. You have to discuss the limitations of the model, when it can be used and when not.

With this data set the initial challenge was dealing with memory issues. The entire data set was about 30 GB, so running data cleaning and feature engineering using a single computer was not an option. Initially we tried to open the files via SQL server and the files would not open only brought up errors. I tried load files in Jupyter notebook but user logs table loading cause the notebook to crush. On the grounds that the size of the user logs file is large, we tried to split data into chunks unsuccessfully. Finally, as last result we cut the user logs file to 30,000,000 rows. I purchased google colab pro subscription in order to increase the RAM size. Instead of using SQL queries and with your approval I created the features inside the python notebook.

A major part of the work for this project was feature engineering through python. I added intuitive features like discount and cost per day. I paid close attention to transformed data.

One of the challenges with this dataset is that it With a churn rate of approximately 12%, there is a moderate class imbalance issue.We defined imbalanced data as smaller/equal to threshold of 10%. Since 12% is above threshold, we will not use over-sampling technic but on trained a model with class weights in the finetuning model stage.

I deployed three different binary classification models to predict whether users will churn or not: Logistic regression, decision tree, random forest, KNN, Adaboost and XGBoost

The accuracies of the method were compared and XGBoost had best prediction performance based on AUC measurement.

For xgboost I used 5 parameters 'max\_depth': [6,7,8,9], 'n\_estimators': [50, 100], 'objective': ['binary:logistic'],'subsample': [0.8], 'scale\_pos\_weight': [scale\_pos\_weight], 'seed': [123] and ran 80 fits at total. We achieved approvement of 2.54%.

XGBoost (as with other boosting techniques) is more likely to overfit than bagging does (i.e. random forest) but with a robust enough dataset and conservative hyperparameters, higher accuracy is the reward. XGBoost takes quite a while to fail, that's another drawback when compared to more naive approaches. Overall though, as far as boosting goes, XGBoost is an upgrade on an idea (gradient boosting) that was itself an improvement on naive bagging techniques. Because it was created relatively recently and its design considered the issues with existing models, it tends to outperform them based on those metrics. It is important to remember that XGBoost is essentially just regular gradient boosting with some regularization and such (which provides some help in avoiding overfitting), so any set of circumstances which cause gradient boosting to fail could cause XGBoost to fail.

# Data retrieval protocol

Attached as separated excel file.