Universidad del Desarrollo
Universidad de Excelencia

Modelo Predictivo de Fuga de Clientes en Servicio de Streaming
UDD MAGÍSTER DE DATA SCIENCE

José Pedro Cordero / Hernán Rivera

Desafío

- **KKBOX** es el servicio de transmisión de música líder en Asia, y cuenta con la biblioteca de música Asia-Pop más completa del mundo con más de 30 millones de canciones.
- El desafío consiste en desarrollar un algoritmo que prediga si un usuario de suscripción dejará de usar el servicio, esto, con un conjunto de testeo (Marzo 2017) de la empresa **KKBOX**.
- Para un negocio de suscripción, predecir con precisión el abandono es fundamental para el éxito a largo plazo. Incluso pequeñas variaciones en la rotación pueden afectar drásticamente las ganancias.
- La métrica de evaluación es **Log Loss**, donde **N** es el número de observaciones, **Log** es el logaritmo natural, **yi** es el objetivo binario y **pi** es la probabilidad de predecir que **yi** es igual a 1

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i))$$

Definición

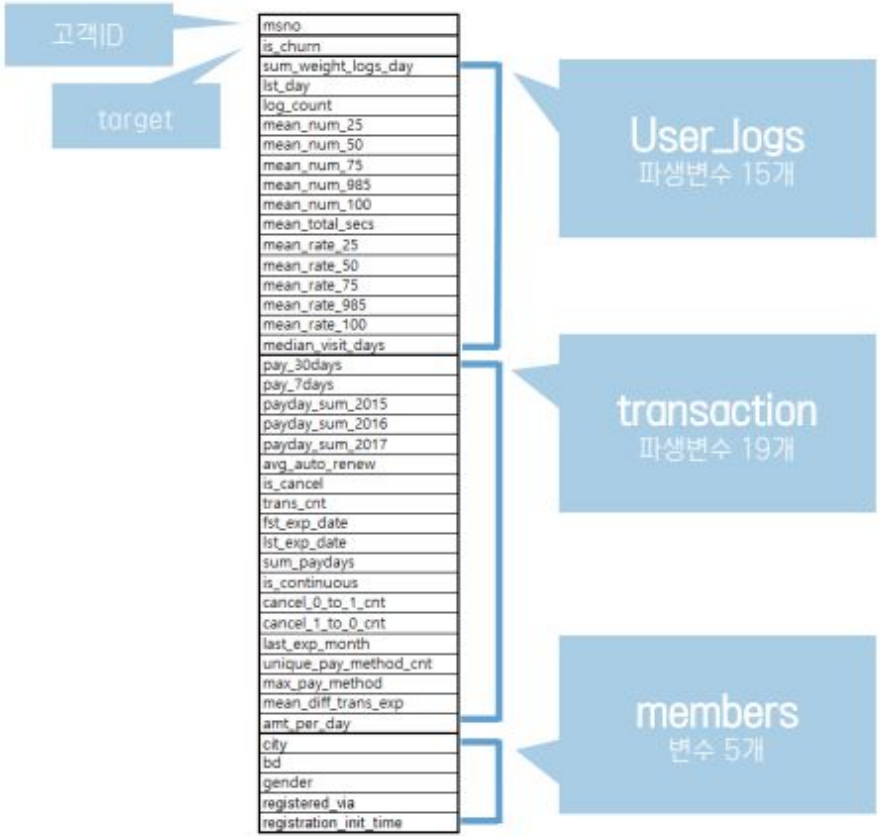
- **Churn (Wikipedia)**: La tasa de abandono, cuando se aplica a una base de clientes, se refiere a la proporción de clientes contractuales o suscriptores que dejan un proveedor durante un período de tiempo determinado.
- En contexto, la definición de **abandono/renovación** puede ser complicada debido al modelo de suscripción de KKBox. Dado que la mayoría de la duración de la suscripción de KKBox es de 30 días, muchos usuarios se vuelven a suscribir cada mes. Los campos clave para determinar la **rotación/renovación** son la fecha de la transacción, la fecha de vencimiento de la membresía y el campo **is_cancel**.
- El campo **is_cancel** indica si un usuario cancela activamente una suscripción. La cancelación de la suscripción no implica que el usuario haya cancelado el servicio. Un usuario puede cancelar la suscripción del servicio debido a cambios en los planes de servicio u otros motivos. **El criterio de "abandono"** no es una nueva suscripción de servicio válida dentro de los 30 días posteriores a la expiración de la membresía actual.

Dataset

El conjunto de training y testing corresponde a los usuarios cuya membresía en el mes de febrero de 2017, y los datos de testing corresponde a los usuarios cuya suscripción expiraba en el mes de marzo de 2017.

Datasets	Descripción	Número de Registros
members_v3.csv	Información de miembros	~ 5m de registros
train.csv train_v2.csv	Dataset de entrenamiento. Dataset de entrenamiento actualizado (**Marzo 2017**)	**~** 22 m de registros
transactions.csv transactions_v2.csv	Transacciones de usuarios hasta **2/28/2017** Transacciones de usuarios hasta el 31/03/2017 (Actualizado el 11/06/201)	
user_logs.csv user_logs_v2.csv	Log diario que describe el comportamiento de los usuarios, hasta el **28/02/2017** Log diario de usuarios hasta **3/31/2017**	**~** 400m de registros
sample_submission_zero.csv sample_submission_v2.csv	Muestra que contiene el user id Muestra actualizada	

Modelo de Datos



Modelo de Datos

El modelo de datos de la competencia KKBOX se divide en dos tipos de tablas, las que contienen el total de registros y las que contienen una sampleo del total de registros para efectos de poder procesarlas con poco poder de cómputo, estas tablas finalizan en _v2 .

train
msno
is_churn

train_v2
msno
is_churn

sample_submission_v2
msno
is_churn

transactions
msno
payment_method_id
payment_plan_days
plan_list_price
actual_amount_paid
is_auto_renew
transaction_date
membership_expire_date
is_cancel

transactions_v2
msno
payment_method_id
payment_plan_days
plan_list_price
actual_amount_paid
is_auto_renew
transaction_date
membership_expire_date
is_cancel

user_logs
msno
date
num_25
num_50
num_75
num_985
num_100
num_unq
total_secs

user_logs_v2
msno
date
num_25
num_50
num_75
num_985
num_100
num_unq
total_secs

members_v3
msno
city
bd
gender
registered_via
registration_init_time

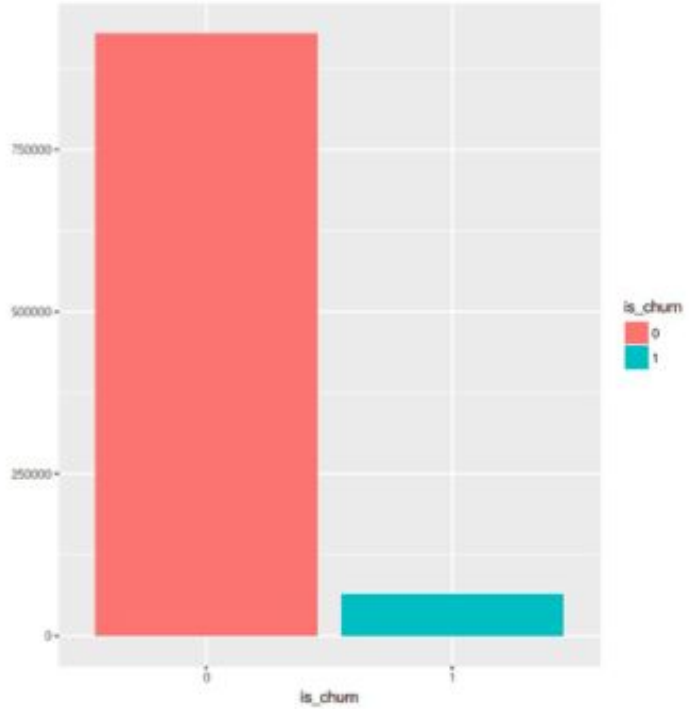
Desafíos Técnicos

Dataset muy grande:

- user_log.csv tiene 30 GB, 2 mil millones de registros

La etiqueta a predecir está desbalanceada:

- (churn / no churn) = 1: 15.5
- La precisión no es confiable (predecir “no churn ” equivale al 94% de la precisión)



Outliers

- Muchas features tienen outliers:
- Birthday: usuario de 1.000 años
- Total secs: valores negativos
- Date: fechas fuera del ámbito de estudio
- Gender: información faltante

Modelamiento e hiperparámetros

- Extreme Gradient Boosting Trees (**XGBoost**): es una implementación de código abierto del algoritmo “gradient boosted trees”, un meta-algoritmo cuyo objetivo es convertir una colección "week learners" en un modelo robusto.
    - hiperparámetros
        -

Métricas de evaluación y conclusiones

Las métricas para evaluar la performance de model:

- log-loss

Impacto de cada feature en el modelo, de dos maneras:

- Tests KS univariados comparando las poblaciones Churn vs No-Churn
- Performance del modelo entrenado con y sin el feature (sensibilidad)

El uplift del resultado final para el segmento de los top 10K msno's. (%churn rate top 10K vs %churn rate total), es de: XXX

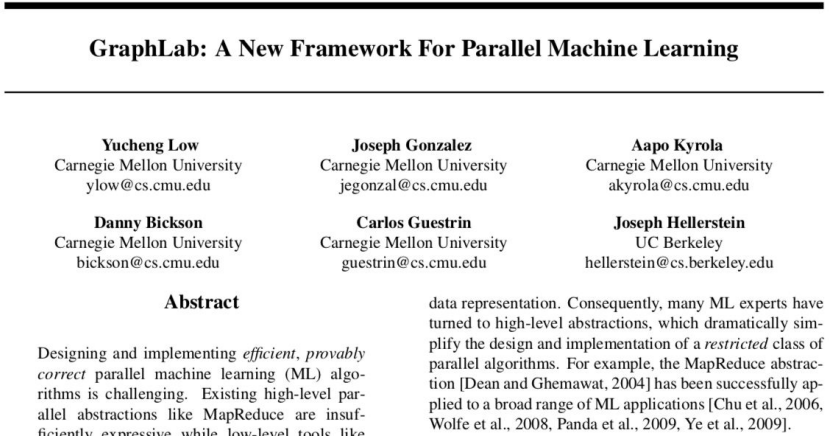
Conclusiones:

- Los parámetros del modelo XGBoost final fueron seleccionados de forma heurística, estos se pueden mejorar aún más spliteando el conjunto de entrenamiento y testeo, y "cross validation" para el ajuste de parámetros.
- Con más tiempo podríamos tener una comprensión más amplia de los hiperparámetros y de los mecanismos detrás de XGBoost para que podamos utilizar mejor esta poderosa herramienta.

Implementación

Para abordar el análisis exploratorio de de los datos utilizaremos el siguiente ambiente:

- Notebook Mac Air 8 GB RAM
- Disco Duro SATA 512 GB
- Framework Python Graphlab de Turi



GraphLab: A New Framework For Parallel Machine Learning

Yucheng Low
Carnegie Mellon University
ylow@cs.cmu.edu

Joseph Gonzalez
Carnegie Mellon University
jegonzal@cs.cmu.edu

Aapo Kyrola
Carnegie Mellon University
akyrola@cs.cmu.edu

Danny Bickson
Carnegie Mellon University
bickson@cs.cmu.edu

Carlos Guestrin
Carnegie Mellon University
guestrin@cs.cmu.edu

Joseph Hellerstein
UC Berkeley
hellerstein@cs.berkeley.edu

Abstract

Designing and implementing *efficient*, *provably correct* parallel machine learning (ML) algorithms is challenging. Existing high-level parallel abstractions like MapReduce are insufficiently expressive while low-level tools like

data representation. Consequently, many ML experts have turned to high-level abstractions, which dramatically simplify the design and implementation of a *restricted* class of parallel algorithms. For example, the MapReduce abstraction [Dean and Ghemawat, 2004] has been successfully applied to a broad range of ML applications [Chu et al., 2006, Wolfe et al., 2008, Panda et al., 2009, Ye et al., 2009].

Implementación

En cuanto a las transformaciones de los datasets BigQuery de GCP, cuyas principales características son las siguientes:

Google BigQuery

- Sistema para Data warehousing autoadministrable
- Base de datos columnar
- Serverless (no-ops)
- Soporta standard SQL legacy SQL.
- Las consultas pueden escalar a cientos de CPU´s por medio de muchos nodos de cómputo.
- Sirve tanto para almacenamiento como análisis de Datasets.