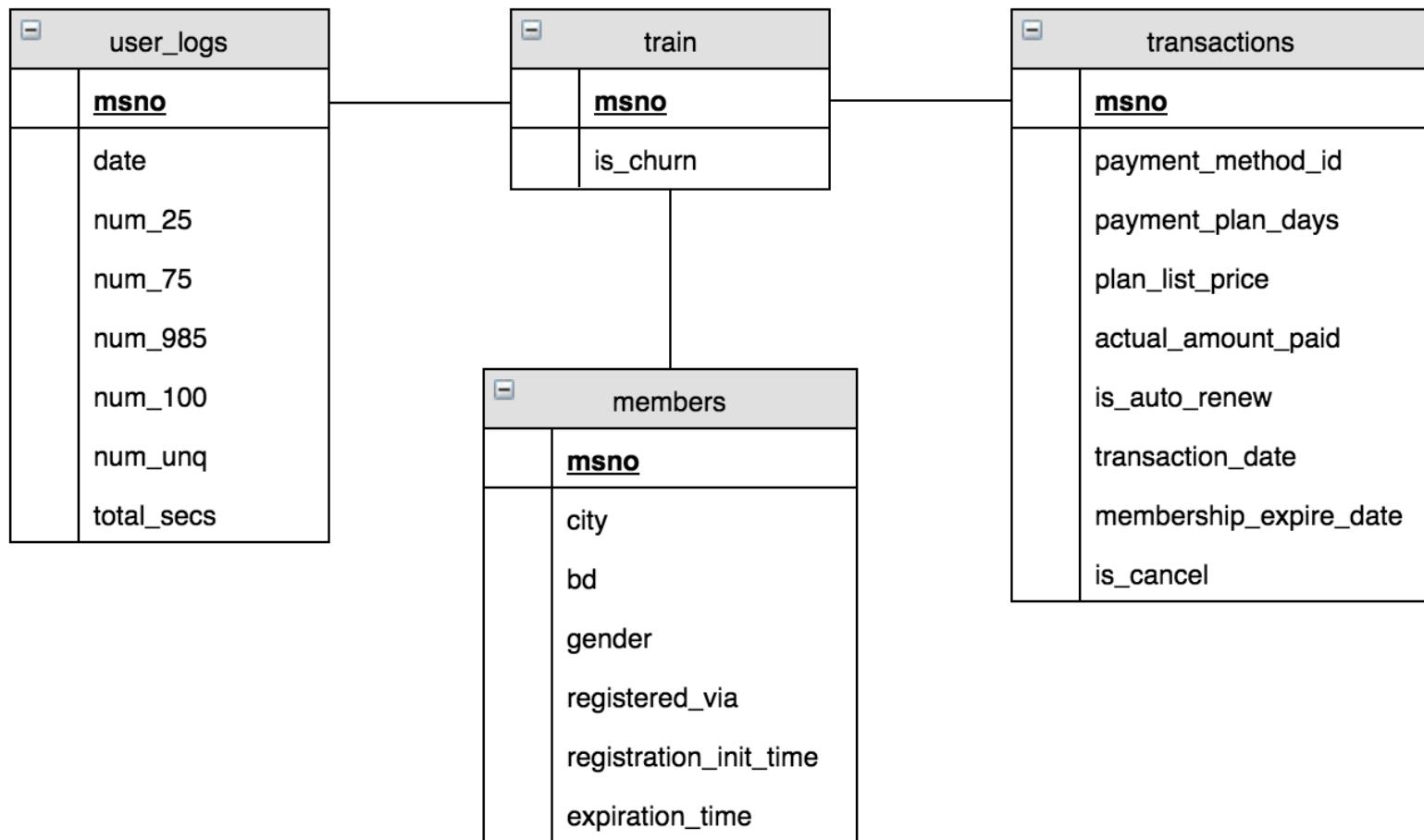


# KKBox's Churn Prediction Challenge

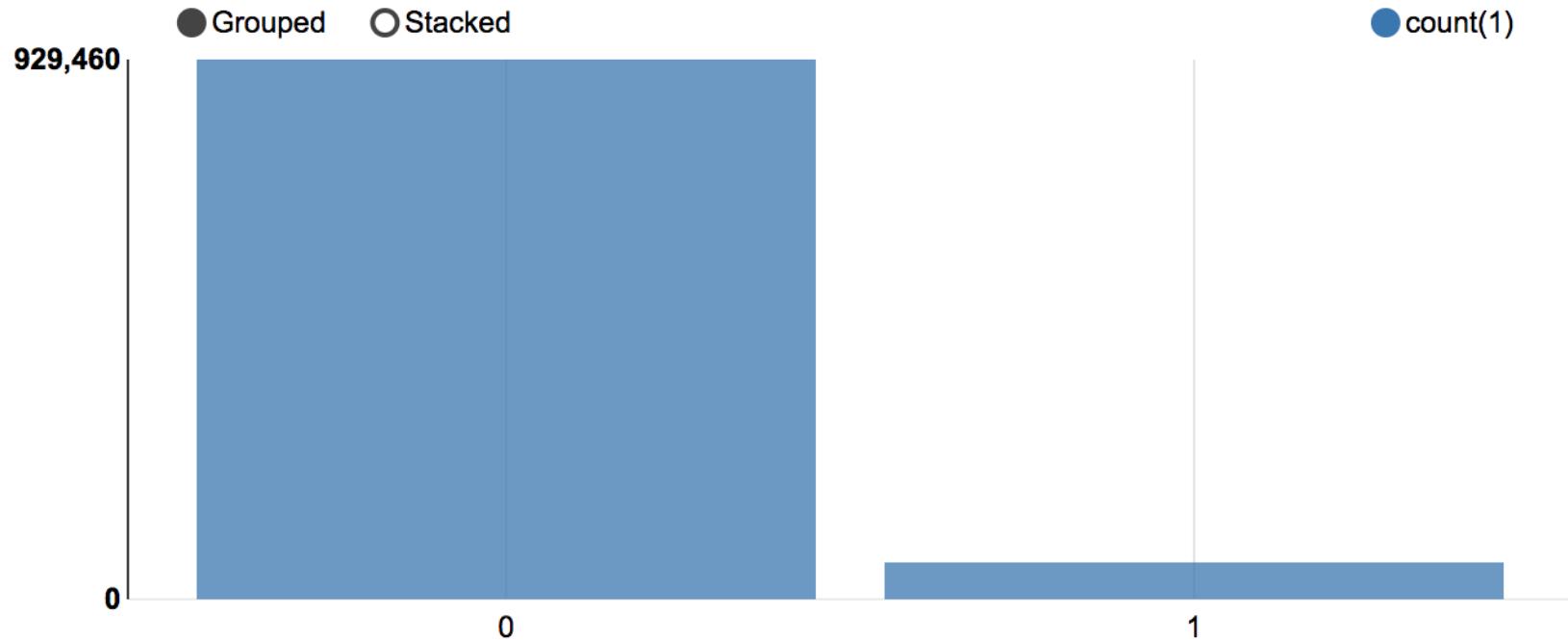
Can you predict when subscribers will churn?



# 데이터 소개



# 데이터 소개



- 극단적인 불균형 데이터
- 이탈 비율 :  $929,560 / 63,471$
- 나중에 알게된 사실 : 테스트셋은 이탈유저 1명 (?)

# 데이터 소개

- Kaggle에서 보기 힘든 대용량 데이터
- train : 992,931
- transaction : 21,547,746
- member : 5,116,194
- userlog : 392,106,544 (약 4억)
- Pandas로 읽으려 하면, "MemoryError" 발생

# Spark을 활용한 대용량 데이터 처리

- 데이터의 사이즈를 줄이고 pandas chunk, map, concat 등을 이용하면 어떻게든 읽을 수는 있으나, 전처리까지 하려면 한참 걸림
- 그냥 PySpark으로 처리하자
- 쉽고 빠르게 분산처리를 지원
- pandas와 API가 유사하면서 완벽하게 호환, toPandas()

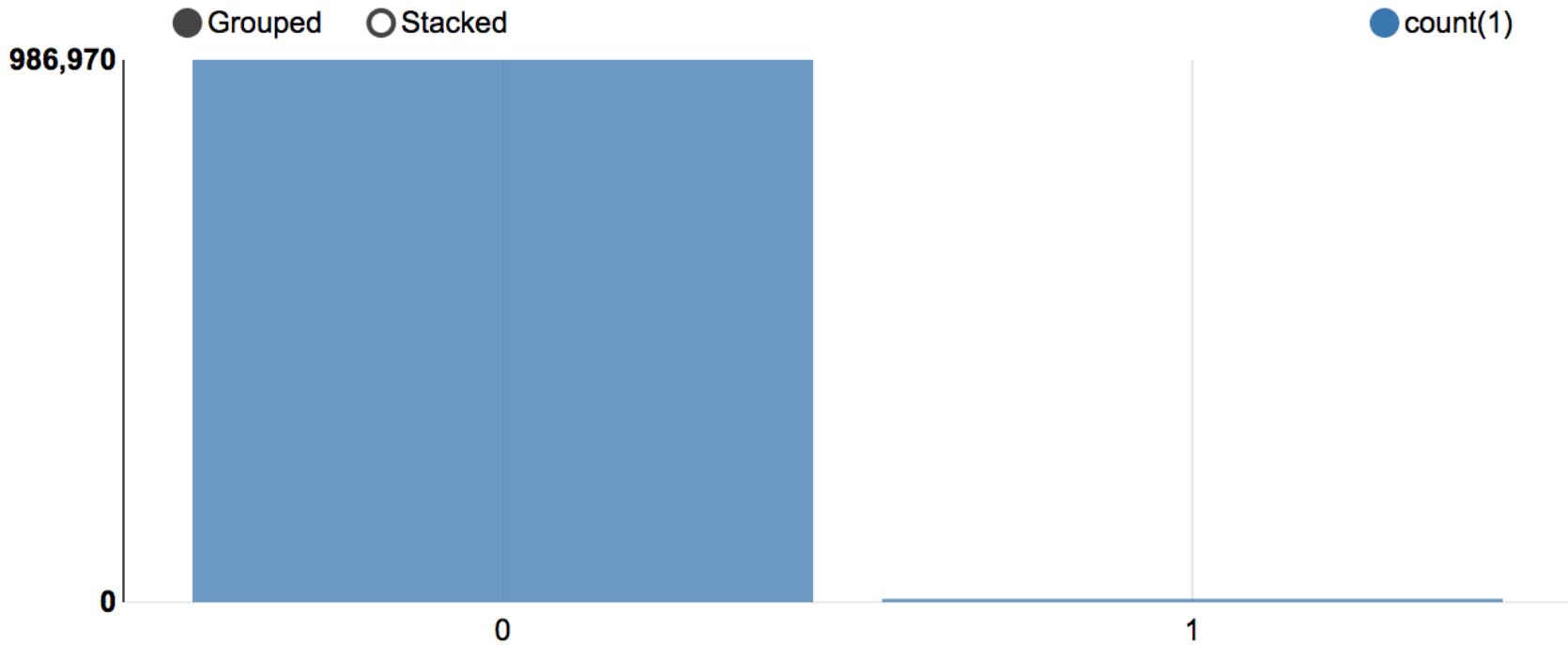
# Spark SQL + Zeppelin EDA



# 유저 이탈 정의

- 트랜잭션이 발생하는 경우 : 실제 결제, 자동 결제 설정, 취소
- 25일까지 구독을 적극적으로 취소
- 이후 30일 동안 거래가 갱신되지 않는 유저
- training : 17년 2월 이탈 유저
- test : 17년 3월 이탈 유저
- 실제 제공된 데이터의 기간은 2015년 부터 2월 까지
- 그렇다면 과거 데이터를 통해 이탈자를 생성할 수 있지 않을까?

# 1월 이탈자 데이터 생성

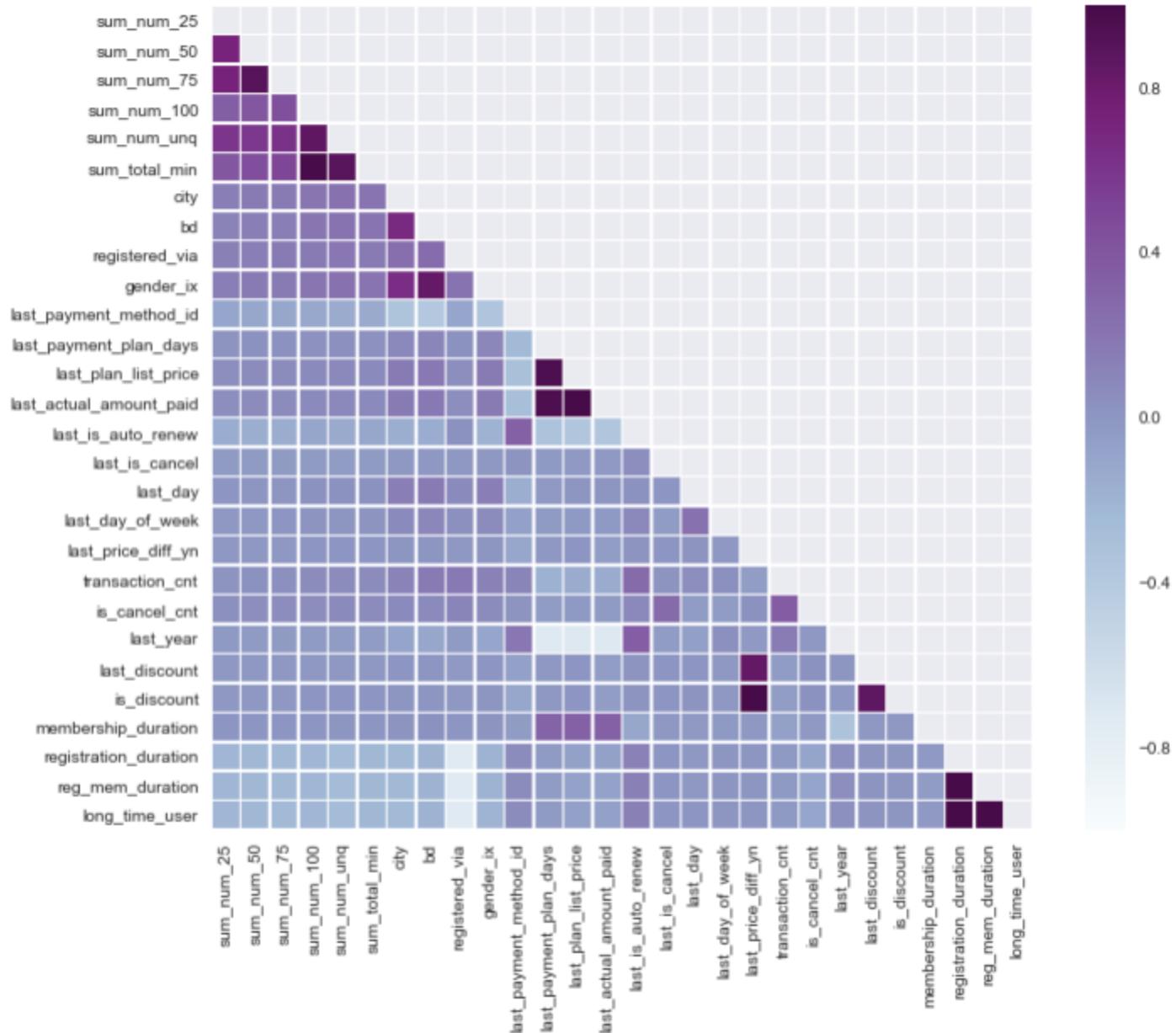


- train, member에 정의되어 있는 유저 아이디를 이용
- 이탈 비율 :  $986,970 / 5,961$
- 안타깝게도 별로 효과가 없었음

# 사용한 Feature

- 유저의 히스토리 (최근 6개월, 1년...)
- 가장 마지막으로 남긴 로그
- 음악을 듣는 간격, 날짜 전처리
- 할인 여부, 장수 유저인지, 지불 금액대비 얼마나 듣는지
- 이후에 열린 추천 대회로부터 얻은 노래 장르
- 기타 등등...

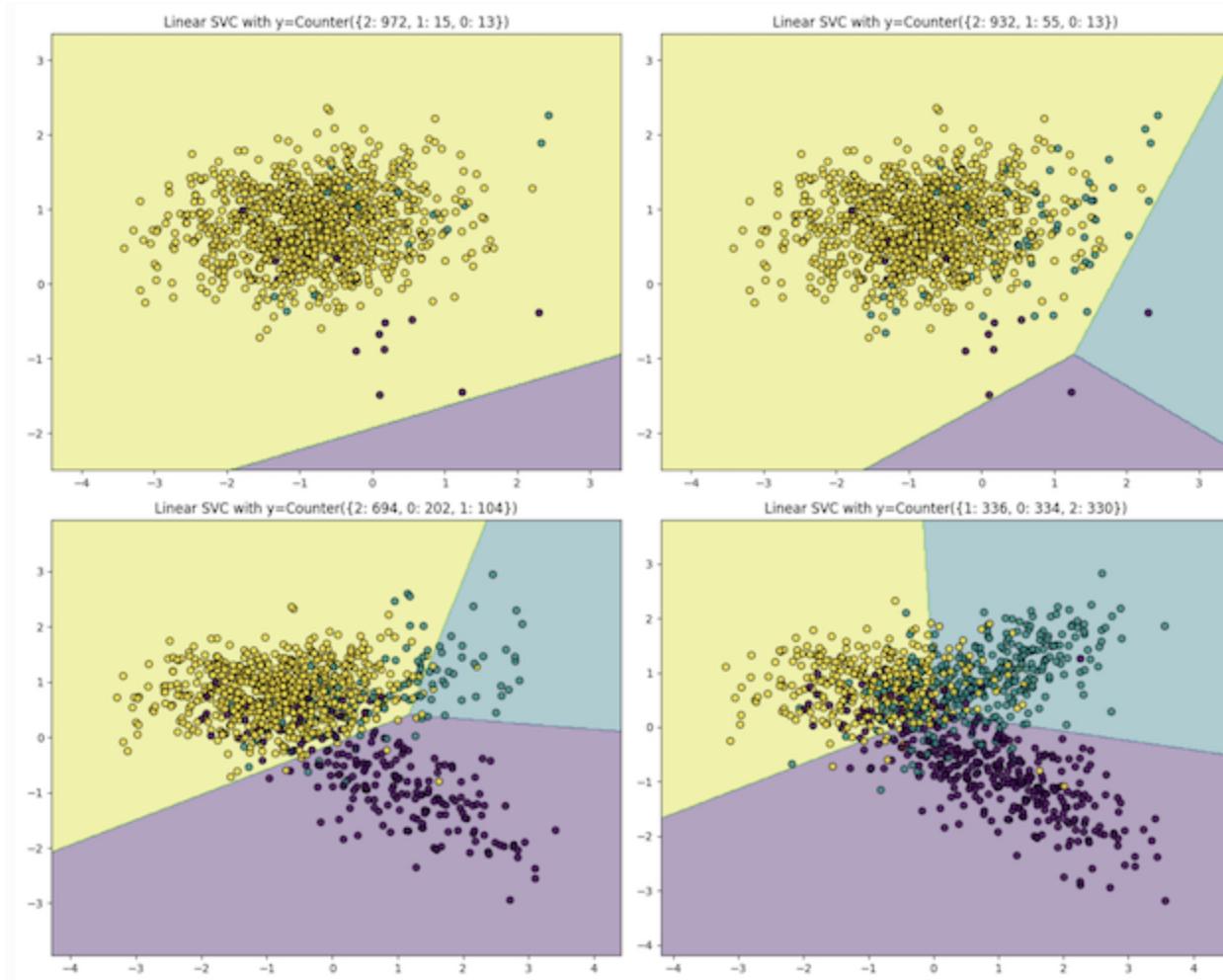
# Feature Selection



# Feature Selection

- RandomForest 모델의 경우, Recursive Feature Elimination을 사용
- `sklearn.feature_selection.RFE`
- `metrics.log_loss`값과 Kaggle 리더보드의 log loss 값이 달라 변수를 하나씩 추가, 삭제해가면서 변수를 선택함

# 데이터 불균형 처리



<http://contrib.scikit-learn.org/imbalanced-learn/stable/index.html>

## Over-sampling

- Naive random over-sampling(RandomOverSampler)
- From random over-sampling to SMOTE and ADASYN(SMOTE, ADASYN)

## Under-sampling

- Prototype generation(ClusterCentroids)
- Prototype selection
- Controlled under-sampling techniques
- RandomUnderSampler, NearMiss
- Cleaning under-sampling techniques
- AllKNN, InstanceHardnessThreshold

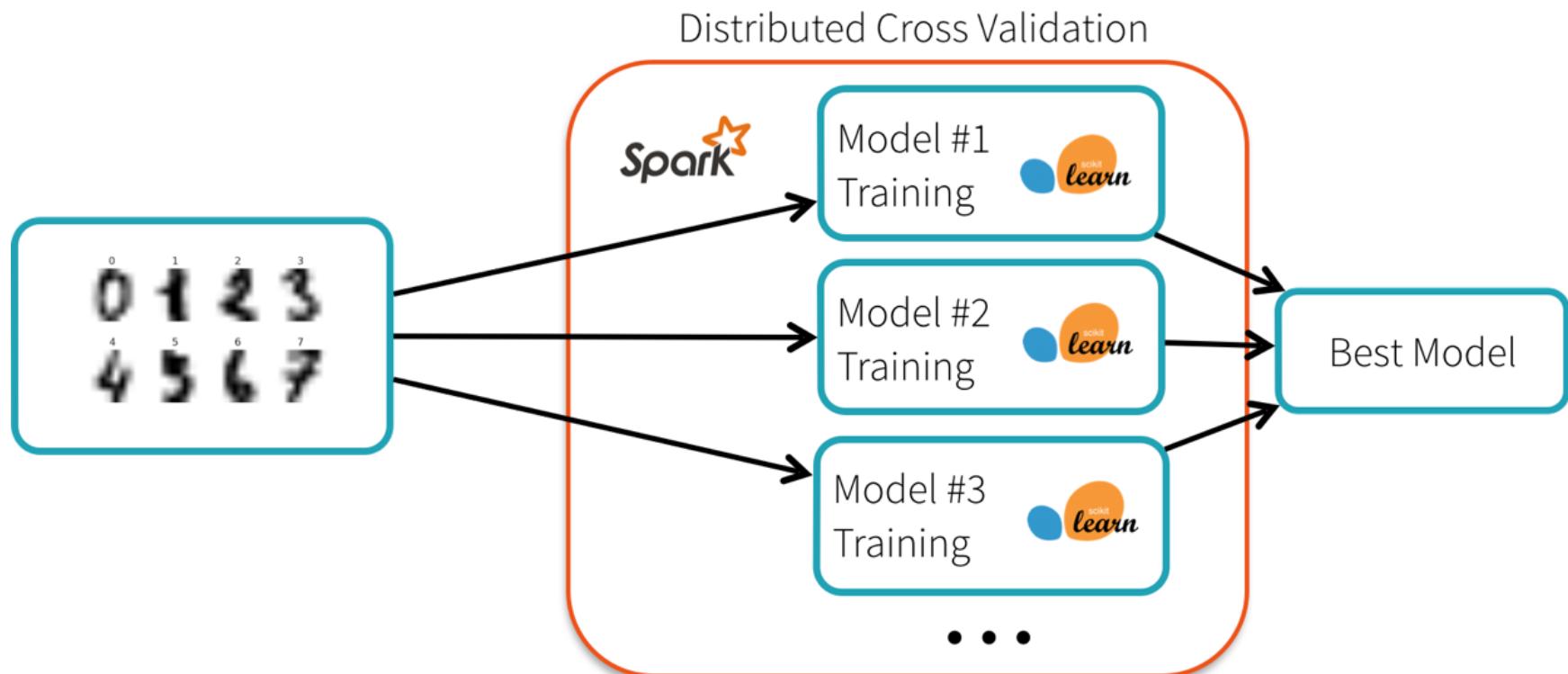
# 데이터 불균형 처리

- 모델의 파라메터 튜닝을 통해 해결하는 방법
- StratifiedKFold + Model
- XGBoost : scale\_pos\_weight, max\_delta\_step
- LightGBM : is\_unbalance

# Parameter Tuning

- GridSearchCV (30시간 돌리다 포기)
- RandomizedSearchCV (log\_loss값이 default 보다 안좋음)

# Distributed Parameter Tuning



- `from spark_sklearn import GridSearchCV`
- Sklearn에서 사용하던 것과 동일한 인터페이스 사용
- AWS EMR 환경 + m4.xlarge 10대에서 약 50분 소요

# Stacking

- XGBoost (scale\_pos\_weight로 데이터 불균형 조정)
- LightGBM (is\_unbalance로 데이터 불균형 조정)
- RandomForest (결과 값이 더 안 좋아짐 ㅠㅠ)

# 현재 리더보드 상황

#	△1w	Team Name	Kernel	Team Members	Score ⓘ	Entries	Last
1	▲ 20	<b>Henry Lu</b>			0.00000	19	2d
2	new	<b>polyqrf</b>			0.00000	2	2d
3	▲ 170	<b>jxlijunhao</b>			0.00000	15	2d
4	new	<b>ZiyangChan</b>			0.00000	4	2d
5	new	<b>ZoeGreen</b>			0.00000	3	2d
6	▲ 114	<b>gjwei</b>			0.00000	6	2d
7	▲ 21	<b>Random String</b>			0.00000	51	2d
8	new	<b>ololo</b>			0.00000	2	2d
9	new	<b>Zeljko is a cutie</b>			0.00000	9	11m
10	▲ 175	<b>Vadim Borisov</b>			0.00000	8	5h
11	▲ 104	<b>sipgeone</b>			0.00000	4	2d
12	new	<b>DmitryTsesarev</b>			0.00000	1	2d
13	▲ 129	<b>Nicolas</b>			0.00000	20	2d
14	new	<b>shu_t</b>			0.00000	1	2d
15	new	<b>Artem.Sanakoev</b>			0.00000	1	2d
16	▲ 86	<b>TenYun</b>			0.00000	10	2d

# 최종 결과 및 느낀점

- 모델을 돌리기에 앞서 데이터셋을 잘 추출하는게 얼마나 중요한지 깨닫게 됨 (대회 주최자 욕 바가지로 드심)
- 분산처리는 강력함 (전처리, 파라메터 튜닝)
- 불균형 데이터의 데이터 균형화가 중요
- parameter tuning보다 feature engineering으로 로그 로스 값이 많이 줄어듬
- 로그 데이터 많다고 더 좋은 결과 값을 가져오는 것은 아님
- Data leakage로 대회는 망하고 허탈함
- Kaggle에서 리서치 대회 참가는 비추

Thank you