머신러닝

조원삼, 나요셉, 김진호

Kaggle 분석보고서

CONTENTS

1. 1round 분석과정

1. Data Cleansing

panel

1) 결측치 처리

- BIRTH , GENDER , REGION : 최빈값

2) 질문 별 전처리 적용 법

- Single answer : 단일응답 중 질문 선다에 없는 값(error값) : -1
- Multi answer : 단일응답 중 질문 선다에 없는 값 error값 : -1 / 다중응답 : -2

survey

- CATEGOIRES : 사용하기 애매하다고 판단 – 열 차제 제거

Data Cleansing을 하며 userID, surveyID 각각을 기준으로 Feature를 만들기로 실험계획을 수립함

-> surveyID가 train 과 test 각각 동일하지 않은 값들이 많아 surveyID는 고려하지 않기로 판단

2. Feature Engineering

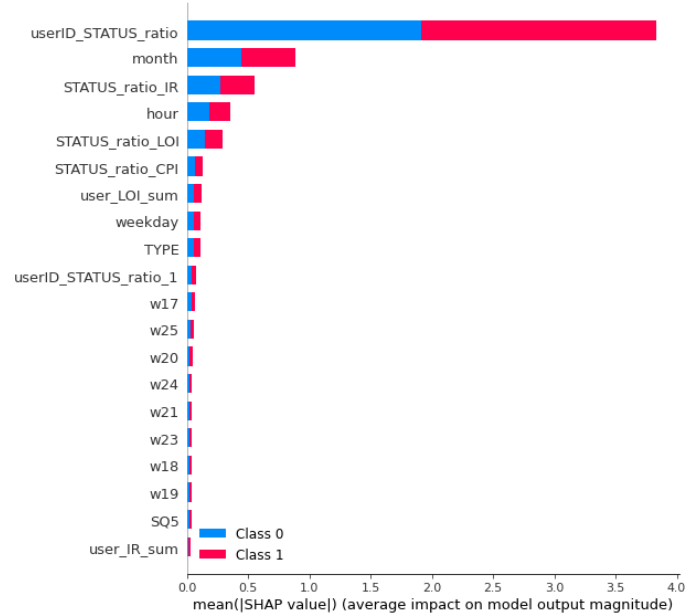
```
Index(['TYPE', 'SQ2', 'SQ3', 'SQ4', 'SQ5', 'SQ6', 'SQ7', 'SQ8', 'A1', 'B3',
       'B4', 'B5', 'C1', 'C2', 'C3', 'F1', 'F2', 'H1', 'T1', 'X1', 'X3', 'X4',
       'DQ1', 'DQ2', 'DQ3', 'DQ4', 'DQ5', 'DQ6', 'DQ7', 'STATUS_ratio_IR',
       'STATUS_ratio_LOI', 'STATUS_ratio_CPI', 'weekday', 'hour', 'month',
       'TITLE_S=해외', 'TITLE_S>0.4', 'TITLE_S1>0.3', 'user_IR_sum',
       'user_LOI_sum', 'user_CPI_sum', 'userID_STATUS_ratio', 'SQ4best3',
       'SQ5best3', 'SQ6best3', 'SQ7best3', 'SQ8best3', 'A1best3', 'B3best3',
       'B4best3', 'B5best3', 'C1best3', 'BIRTH_STATUS', 'w17', 'w18', 'w19',
       'w20', 'w21', 'w22', 'w23', 'w24', 'w25', 'w26',
       'userID_STATUS_ratio_1'],
```

Data Leakage를 감안하고 Target값인 STATUS를 사용하여 피쳐를 만들기로 결정

-> 구매추이, user별 응답률, STATUS 비율이 높은 질문 3개 응답여부 등

-> IR, CPI, LOI 구간별 가중치를 부여하여 각각의 값 별, 구간 별 응답률을 계산

3. Feature Selection



각 피쳐들의 영향력을 보고자 하였고, 다양한 시각화 패키지를 통해 A/B test를 효율적으로 하기 위해 Shap를 선택

LGBM 기반 Shap를 통해 각 피쳐들의 영향력을 파악 후 importance < 0 인 피쳐들은 제거

- 예상대로 STATUS를 사용한 피쳐들의 영향력이 압도적으로 높게 나옴
- 예상치 못한 month의 높은 영향력으로 시계열 피쳐를 더 만들고자 함

< Feature Selection을 통한 Insight >

- Status를 사용하되, Data Leakage를 최대한 해결해야 함.
- 시계열 피쳐 또한 train-test간의 기간차로 인한 과적합이 일어날 수 있음을 예상
- Key Feature 몇 개만 찾으면 드라마틱한 성능 향상을 기대할 수 있겠다고 예상

4. Modeling

```
from lightgbm import LGBMClassifier

lgbm_wrapper = LGBMClassifier(n_estimators = 400,random_state=42) # 400번 학습

lgbm_wrapper.fit(X_train, y_train, early_stopping_rounds=10, eval_metric="logloss",
                 eval_set=evals, verbose=True)
lgbm_preds = lgbm_wrapper.predict(valid_X)
lgbm_pred_proba = lgbm_wrapper.predict_proba(valid_X)[:,1]
```

```
[400]   valid_0's binary_logloss: 0.35595       valid_0's binary_logloss: 0.35595
```

< LGBM 조기중단을 통한 Insight >

400번 학습할 동안 Loss 값이 떨어지지 않음

- **대용량의 데이터로 학습량이 부족하다고 판단되어 1000번으로 수정하여 모델 학습**
- **Loss값을 더 좋아지지만, 오히려 제출결과는 하락**

-> 적절한 n_estimator 값을 찾아야겠다고 판단

-> 데이터의 크기, 효율적인 모델링, 시간절약을 위해 분류기에서 자주 쓰이는 LGBM 조기중단을 사용

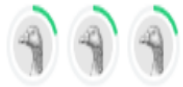
5. Scoring

< Local Score >

```
Confusion Matrix
[[873852  99128]
 [137207 361178]]
accuracy: 0.8394, precision: 0.7846, recall: 0.7247, f1: 0.7535, roc_auc: 0.9100
```

< Kaggle Score >

나요셉+김진호+조원삼



0.88441

0.02559 정도의 성능차이를 보임

-> Data Leakage를 감안하고 Status를 사용하였기 때문에 당연한 결과라고 생각함 (과적합의 늪에 빠질것이라 예상)

2. 1round 후 문제점 분석 및 해결 초점

1round 후 문제점 및 해결 초점

1. 1round 후 문제점

- STATUS 피쳐 활용에 따른 Data Leakage 현상 고도화 (과적합)

실제로 shap를 통해 볼 수 있듯, STATUS 관련 피쳐가 성능 향상에 엄청난 영향을 미치고 있으며, 그에 따라 과적합 현상도 심각함

- 500만개에 육박하는 데이터로 인해 피쳐가 많을수록 과적합 현상 고도화

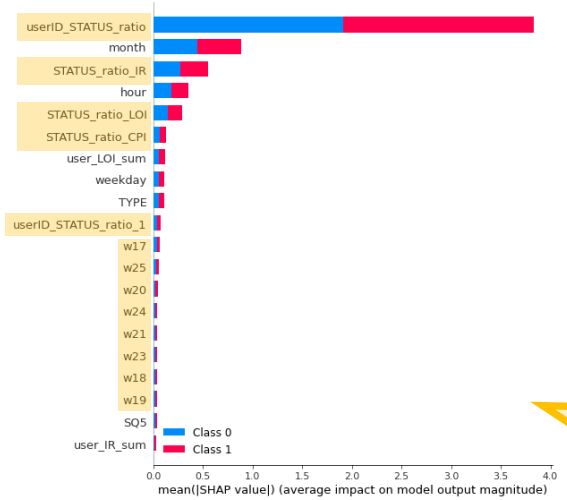
20개 이상만 되어도 1억개의 데이터가 형성되는 것이기 때문에 작은 변화(ex: 파라미터 변경, 피쳐 추가)에도 과적합이 크게 발생하는 현상 발생

Shap를 통해 Feature Selection을 하였지만 과적합이 크게 해소되지 않음

- train-test 간의 기간차이로 인한 Data Leakage 현상 발생

train : 2020-06-01~2021-05-31 / test : 2021-06-01~2021-10-18 기간차로 인한 Data Leakage 현상 발생

Local 과 Kaggle 간의 성능 차이의 가장 큰 원인이라 판단 -> 본 컴패티션에서 가장 중요한 포인트라고 생각



< 상위 20개의 피쳐 중 13개의 피쳐가 STATUS 관련 피쳐 >
-> Data Leakage 현상 고도화 + 성능 향상의 주요 원인
-> '계륵'같은 존재

< train-test 간의 기간차이 >
2021-06-01 을 기준으로 데이터 간의 기간이 다르다
-> train으로 학습을 하면 test에 맞지않는 기간에 학습
-> Data Leakage를 유발하는 주요 원인이라 판단

1round 후 문제점 및 해결 초점

2. 해결 초점

- TARGET 을 활용한 Feature 최소화

Train-test간의 기간차이 외 Data Leakage의 주요 원인이였던 target값 사용을 최소화 하기로 결정

구매추이, IR-LOI-CPI별 응답률, 유저별 응답률과 같은 주요 피쳐들의 과적합 요인을 다른 피쳐로 상쇄시키도록 초기 실험 계획 수정

-> train-test간의 기간차이로 그 효과가 미비하다고 판단되어 train 데이터 자체를 split 하자고 계획 수정

- Train 데이터 split

로컬에서 자체 validation syste의 신뢰도를 확보하기 위해 before_train과 train을 나눠서 userid 관련 sum과 같은 정보를 before_train에서 가져옴

Train의 기간을 test와 동일하게 맞춰주어 validation의 신뢰도를 높이기 위함

< train 데이터 split을 통한 Insight>

Before_train을 1월로 나눴을 때 test에 없는 userID가 5000개 정도 발견

-> 정보의 손실을 최소화하기 위해 before_train 기간을 2020/11월~2021/2월로 설정

-> 이후 validation 그 데이터로 로컬에서 aftesyste의 신뢰도를 더 확보하기 위해 5월 after_train을 나누어 after_train 데이터로 추가 검증을 하려고 실험 설정

train의 데이터 크기가 작아져
절대적인 성능이 낮았고, 결과적으로
과적합 요인이 해결되지 않음

2. 해결 초점

- 기간 차이로 인한 Data Leakage를 해결하기 위해 **자체 validation system을 구축**

train 데이터를 4Fold로 split 하여 자체적으로 validation을 진행

```
from sklearn.model_selection import KFold, TimeSeriesSplit
from lightgbm import LGBMClassifier

from sklearn.metrics import *

n_fold = 4
folds = TimeSeriesSplit(n_splits=n_fold)
splits = folds.split(train)

pred_proba_list = []
for fold_n, (train_index, valid_index) in enumerate(splits):
    print('Fold:', fold_n+1)
    X_train1, X_valid1 = X_train.iloc[train_index], X_train.iloc[valid_index]
    y_train1, y_valid1 = y_train.iloc[train_index], y_train.iloc[valid_index]
    lgbm_wrapper =  LGBMClassifier(n_estimators = 400,random_state=42)
    lgbm_wrapper.fit(X_train1, y_train1,
                     eval_set=(X_valid1, y_valid1), early_stopping_rounds=10, eval_metric="logloss",
                     verbose=False)

    lgbm_preds = lgbm_wrapper.predict(X_valid1)
    lgbm_pred_proba = lgbm_wrapper.predict_proba(X_valid1)[:,1]
    get_clf_eval(y_valid1, lgbm_preds, lgbm_pred_proba)
```

로컬 : 0.8889 / 캐글 : 0.8512

```
Fold: 1
Confusion Matrix
[[150716  36471]
 [ 35881  78143]]
accuracy: 0.7598, precision: 0.6818, recall: 0.6853, f1: 0.6836, roc_auc: 0.8423
Fold: 2
Confusion Matrix
[[137085  40521]
 [ 38412  85193]]
accuracy: 0.7379, precision: 0.6777, recall: 0.6892, f1: 0.6834, roc_auc: 0.8179
Fold: 3
Confusion Matrix
[[127654  42985]
 [ 39098  91474]]
accuracy: 0.7275, precision: 0.6803, recall: 0.7006, f1: 0.6903, roc_auc: 0.8080
Fold: 4
Confusion Matrix
[[246171  12063]
 [ 25955  17022]]
accuracy: 0.8738, precision: 0.5853, recall: 0.3961, f1: 0.4724, roc_auc: 0.7673
```

< 자체 Validation system을 통한 Insight >

각 폴드별 기간이 상이하기 때문에 해당 system을 통해 각 Fold 별 성능을 파악하고 각 기간 별 최적의 피쳐를 찾을 수 있을 거라 기대

-> 수 많은 시도를 해봤지만, 처음에 의도했던 각 기간별 최적의 피쳐를 찾기에는 불규칙한 성능 추이를 자주 보였고, A/B test를 통해 기간별 split한 train 데이터에 최적의 피쳐를 찾아도 왜 그 피쳐가 최적의 피쳐인지 원인을 파악하는데에 한계가 존재하였음

Validation System을 통해 유의미한 결과를 도출해내지 못해

Train을 test의 5개월 기간과 동일하게 2021-02-01 부터 split하여

test와 기간을 맞춰주자고 결정

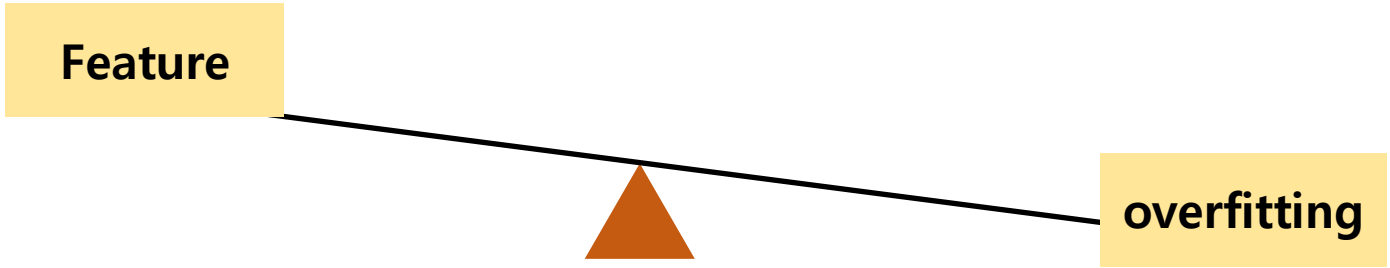
-> Data Leakage를 해소할 수 있는 최소한의 방법이라고 판단

-> before_train(~21/02/01) , after_train(21/02/01~)

3. 2round에 들어가며

- 1round를 경험하여 겪은 과적합, Data Leakage 현상 최소화에 초점
- Feature Set에만 몰두하는 것이 아닌 Catboost, DNN과 같은 다양한 모델들을 사용하고자 함
- 시계열 데이터에 적합한 Fold 기법을 통해 Lucky Feature를 찾고자 함
- 중복을 방지하기 위해 userID 별 응답 비율 제거, 응답추이를 6주로 변경
- IR, CPI, LOI이 중요하다 판단하여 각 수치형 데이터와 관련하여 응답률 별 각 수치형 데이터의 추이에 집중
- 모델 별 다른 Feature Set을 적용 (ex. LGBM : 수치형>범주형, CAT : 수치형<범주형)

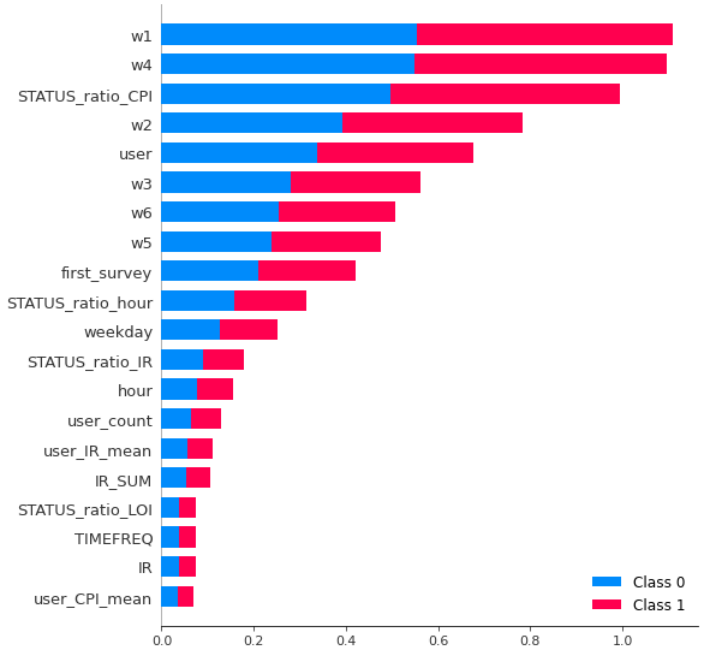
Overfitting을 해결하자!!



3. 2round 분석과정

1. LGBM

- 구매율에 대한 피쳐를 최소화하고 수치형 피쳐로 Feature Set 형성
- Shap importance < 0으로 Feature Selection
- Data Leakage를 최소화하기 위해 before_train에서 피쳐를 생성 후 train에 merge하는 방식 채택
- LGBM 조기중단 사용 (n_estimators = 400)



< Local Score >

```
Confusion Matrix
[[202217  23492]
 [ 21477 103978]]
accuracy: 0.8719, precision: 0.8157, recall: 0.8288, f1: 0.8222, roc_auc: 0.9456
```

< Kaggle Score >

1211LGBM9456.csv 0.89660

5 days ago by

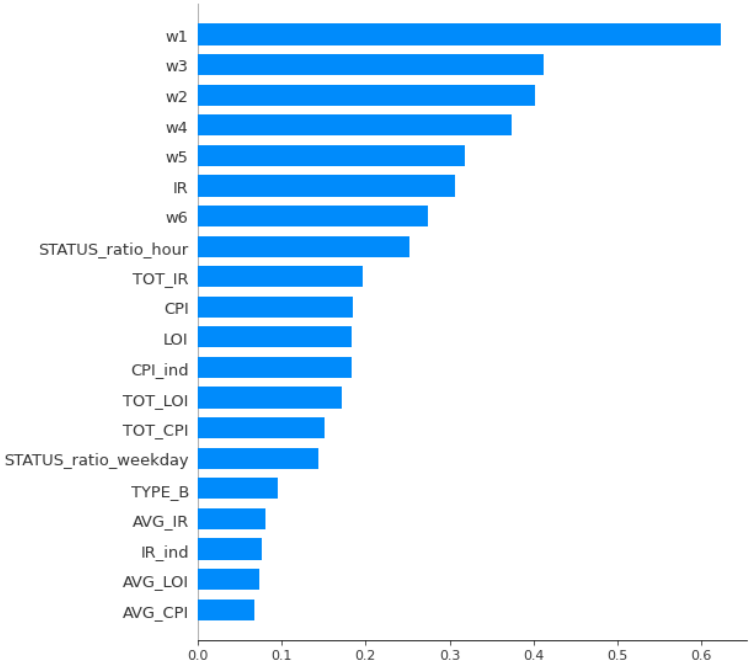
< LGBM 조기중단>

과적합을 잡으려 하였으나 2round에 들어와서 더 심해짐

-> Lucky Feature에만 몰두하여 다른 방법을 찾지 않았던 것이 Fail요소였다고 생각

2. CatBoost

- CatBoost 모델의 특성 상 **범주형 데이터 위주**로 Feature Set 생성
- LGBM에서 우세를 보였던 응답추세만 넣고 IR,LOI,CPI_weight, 총CPI 등과 같은 범주형 피쳐로 구성
- Train와 test의 userID가 유사하다는 것을 생각하여 전체 피쳐를 기반으로 **30개의 군집분석**을 진행
- **Scaling 없이 OneHotEncoding** 진행 -> 대부분 범주형, 수치형 = 비율 -> Scaling 불필요하다고 판단



< Local Score >

```
Confusion Matrix
[[201980  23899]
 [ 21939 103346]]
accuracy: 0.8695, precision: 0.8122, recall: 0.8249, f1: 0.8185, roc_auc: 0.9429
```

< Kaggle Score >

1215_newCatboost_1.csv — 0.89739

2 days ago by

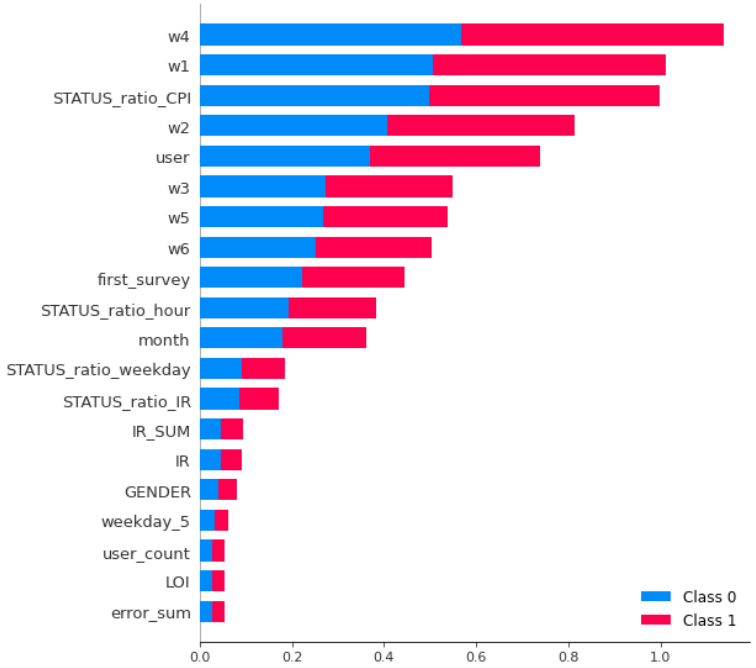
< CatBoost>

LGBM과 마찬가지로 Lucky Feature를 찾는데 몰두하여 제출 9시간 전에 0.895에서 0.897로 성능이 향상됨

-> train 데이터 셋 split을 달리하여 많은 실험을 해봤으면 하는 아쉬움 존재

3. DNN

- LGBM의 Feature Set을 그대로 활용하여 DNN에 적용함
- LGBM 모델에 적합된 피쳐셋이라 DNN에서 드라마틱한 성능 향상을 보여주진 못함
- 타 모델에 비해 DNN에 대한 이해도가 현저히 낮았기 때문에 유연하게 모델을 다루지 못하였음



< Local Score >

```
Confusion Matrix
[[260418  30210]
 [ 26263 134926]]
accuracy: 0.8750, precision: 0.8171, recall: 0.8371, f1: 0.8269, roc_auc: 0.9474
```

< Kaggle Score >

1211DNN2.csv 0.89596

6 days ago by

< DNN >

데이터 자체가 크기에 모델 자체의 레이어나 깊이를 크게 조절하기 힘든 문제

전통적인 머신러닝 모델보다 낮은 성능이 나온 점에서 역량 부족을 느낌

4. Model Tuning

- LGBM의 Feature Set을 그대로 적용하여 RandomForest, XGBoost Tuning 실시
- LGBM의 피쳐셋이 수치형이 가장 많고 비슷한 Tree 계열 모델링에 적합하다고 판단되었기 때문
- MLP에 대해서도 모델링을 진행하였으나, 0.880대에 성능이 분포되어 사용하지 않기로 결정
- LGBM 튜닝의 경우 성능이 최고점이 아니였을 당시에는 유의미하였으나, 0.89660이라는 최고 성능의 경우, 튜닝한 모델 성능이 조기중단의 모델보다 현저히 떨어지는 것을 파악 -> LGBM Tuning X

< Result of Model_Tuning >

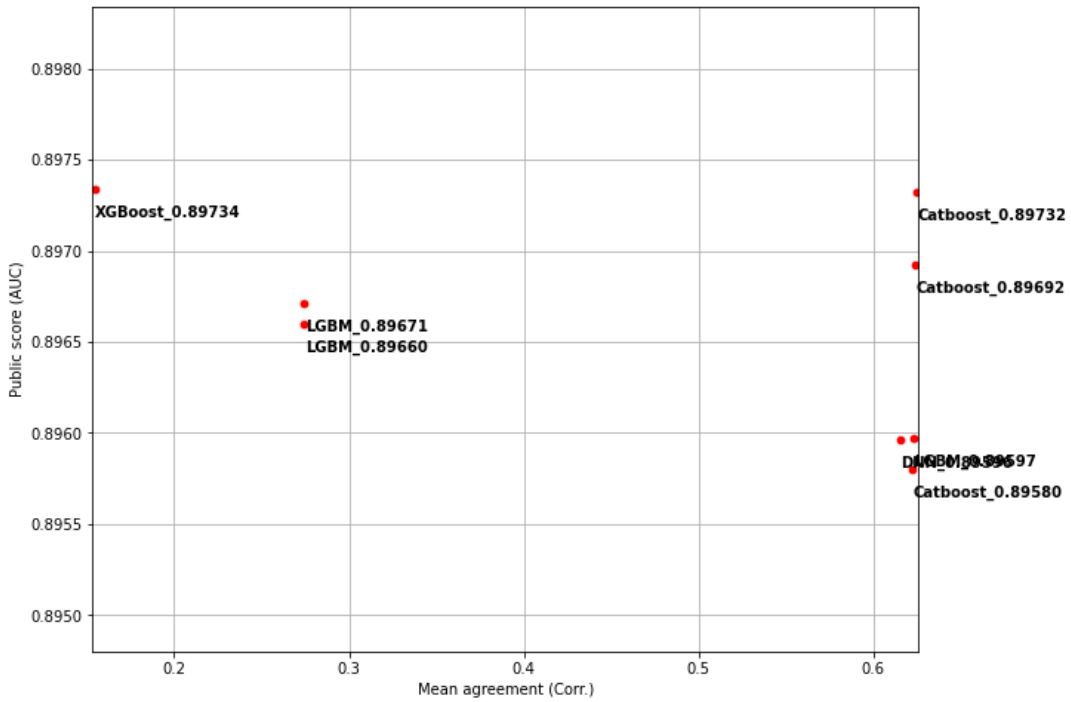
	RandomForest	LGBM	XGBoost	GradientBoost	MLP
Local	0.9129	0.9339	0.9299	컴퓨팅 파워 부족	0.8992
Kaggle	0.8910	0.8892	0.8973	컴퓨팅 파워 부족	0.8801

⇨ Tuning을 통해 XGBoost만 사용

LGBM Tuning의 경우 성능이 더 좋지않은 이유는 train에 편향되게 학습되어
과적합이 더 발생하였다고 판단
-> 현재 사용중인 LGBM 조기중단의 n_estimators의 인자값이 적절하다는 반증

5. Ensemble

	LGBM_1	LGBM_2	LGBM_3	XGBoost	CatBoost_1	CatBoost_2	CatBoost_3	DNN
score	0.89671	0.89660	0.89597	0.89734	0.89732	0.89692	0.89550	0.89596



submissions	Ensemble
전체 p1.5 mean	0.90306
전체 p1.25 mean	0.90335
전체 산술평균	0.90338
Corr 가장 낮은 최고성능 모델 P1.25 mean	0.90253

< Ensemble을 통한 Insight >

각 모델 별 상관관계가 가장 낮은 submission들 끼리의 앙상블을 우선순위로 두었음. 그 결과 0.90253으로 생각보다 높은 앙상블 효과를 가져올 수 있었음.

전체 submissions들의 산술평균의 성능이 가장 높은 것으로 측정됨

-> DNN, Catboost와 같이 상호 이질적인 Feature Set을 적용한 모델들의 성능을 조금 더 높였더라면 조금 더 드라마틱한 앙상블 효과 기대 가능

4. 한계점

한계점

- **특정 성능 이상으로는 과적합이 일어나 성능 향상 X**

- 자체적인 Validation System을 구축하여 과적합을 해결해보려 하였으나 성능 개선 효과 미비
- Validation System으로 각 Fold별 성능이 다르다는 것을 파악 -> 기간 별 성능이 다름 -> 결론적으로 왜 다른지 파악하지 못해 과적합 해결 X
- 좋은 성능이 나온 피쳐에만 집중하여 다른 방식으로 파생되는 피쳐들에 대한 깊이 있는 고려를 하지 못함

- **궁극적인 Data Leakage 현상 해결 X**

- 1round 이후 STATUS 사용을 최소화하고 train-test간의 기간 차이를 최소화하기 위해 train split을 진행하였지만, Data Leakage 해결 X
- 결론적으로 Lucky Feature에 팀원 전원이 국한되어 1round 부터 다양한 실험이 아닌 Feature에 대한 A/B test만을 진행한 것이 실패 요인이라 생각
- 숲을 보지 못하고 나무만 본 것이 결정적인 실패 요인

- **다양한 모델링 기법 적용 X**

- Out of Fold, Bagging, Stacking 등 1round 회의 때 적용하자고 계획했던 모델링 기법들을 사용하지 못함
- 1round 성능이 좋았던 LGBM 조기중단에 국한되어 2round 마감 3일전부터 CatBoost, RandomForest등 다른 모델들의 Feature set을 구성

- **기타 요소**

- 다른 시험기간과 겹쳐 머신러닝을 1순위로 두고 컴패티션에 임했지만 현실적으로 100% 몰두하진 못하였음.
- 제한된 Submission 제출 횟수를 컴패티션 초반 효율적으로 사용하지 못하였음