2019 빅콘테스트
Analysis 분야 챔피언 리그

- 잔존가치를 고려한 이탈 예측 모형 -

2019. 09. 10.

팀명: 투빅스

팀원: 이경택, 안상준, 최영제, 이준걸, 신훈철

NCSOFT®

TOBIG'S

2019 빅콘테스트
2019 BIG CONTEST

AGENDA

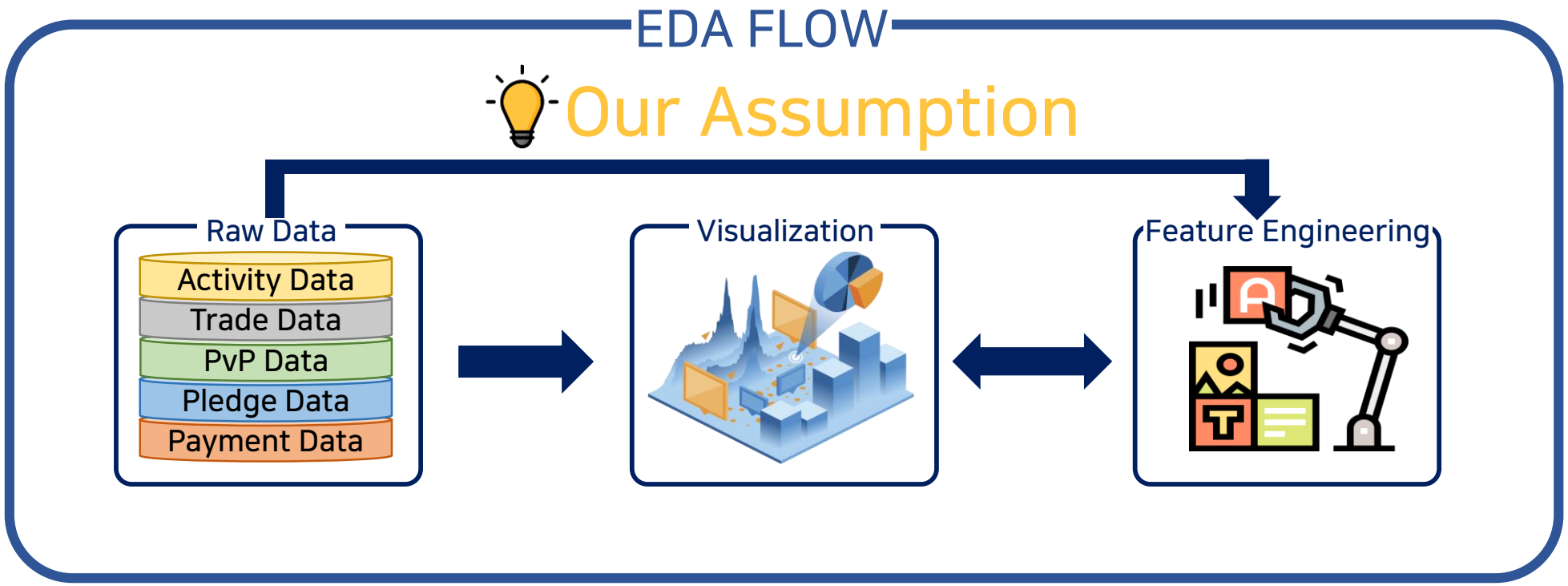
1. 데이터 EDA
2. 모델링
3. SHAP Value
4. 결과 및 해석

AGENDA

1. 데이터 EDA
2. 모델링
3. SHAP Value
4. 결과 및 해석

문제 정의 및 데이터 소개

- 리니지 유저들의 관측된 게임 활동 정보를 활용하여 잔존가치를 고려한 이탈 예측 모형 개발



분석에 앞서 분석의 성능을 높이기 위한 정의한 기본가정을 전제로 파생변수 생성 및 데이터 정제 실시

Activity Data

- 시계열 데이터 정제
  - 각 acc_id에 대한 day 1~28일 데이터를 행을 기준으로 이어 붙여 시계열 데이터의 특징을 반영함.
  - 게임에 참여하지 않은 day의 경우 열의 최솟값으로 고려하여 0으로 Imputation을 진행. 이때 묶이는 day끼리 value는 합으로 진행하였음.

Original Data

day	acc_id	server	playtime	npc_kill	···	Enchant_count
1	75001	aa	1.441844	0.000000		0
1	75001	aa	0.283219	2.247978		0
...	...	...	...	...	...	...
28	72319	bs	1.142240	0.000000		0
28	73739	bs	1.125855	0.000000		0

정제 Data

	day1			day2				day28		
acc_id	playtime	···	Enchant_count	playtime	···	Enchant_count	...	playtime	···	Enchant_count
75001	4			3			...	4		
75001	2			1				2		

그러나 28일에 대한 모든 고려는 Data Dimension이 너무 커지는 단점이 있음.

Activity Data

- 해결 방법
  - Data Dimension이 너무 늘어나기 때문에 28일 전체로 진행했을 경우 Data set이 매우 무거워짐.
  - 따라서 1day, 2days, 4days, 7days(week), 14days(2weeks), 28days(4weeks) 로 기준으로 실험을 진행하여 Split point 기준을 잡고자 하였음.

Case1 (1day)

day1

acc_id	npc_kill	···	Enchant_count
75001	0.00000		
75001	2.73646		

...

day28

npc_kill	···	Enchant_count
0.00000		
2.73646		

Case2 (2days)

2day1

acc_id	npc_kill	···	Enchant_count
75001	0.00000		
75001	2.73646		

...

2day14

npc_kill	···	Enchant_count
0.00000		
2.73646		

...

Case5 (2weeks)

week1 week2

acc_id	npc_kill	···	Enchant_count	npc_kill	···	Enchant_count
75001	0.00000			0.00000		
75001	2.73646			2.73646		

Activity Data

- 실험 방법
  - 1day, 2days, 4days, 7days(week), 14days(2weeks), 28days(4weeks)로 기준으로 Feature를 생성하여 각 Case별로 10-CV로 survival_time를 예측함.(63일 초과 1 / 미만 0)
  - Accuracy와 Data Dimension을 고려하여 Split Point를 설정함.

Case	Accuracy	Dimension Size
1day	0.78	672
2days	0.775	336
4days	0.771	196
7days (week)	0.762	112
14days (2weeks)	0.74	56
28days (4weeks)	0.72	28

성능은 1day가 가장 좋았으나, Dimension을 고려하여 Split point를 2days로 설정

Activity Data

- 파생 변수 생성
  - 1) 접속
    - playtime과 day를 통해 계정이 얼마나 접속, Play를 했는지를 나타내는 변수 생성

Feature Name	Description
week_time1~14	일별(2일합) playtime 비중
total_ply_time	총 play 시간
row_ta	접속 횟수
total_day	접속 일수

  - 2) 서버
    - 서버에 대한 정보로 User가 이용하는 서버 정보를 포함하는 변수생성

Feature Name	Description
svcnt	해당 계정이 접속한 서버의 개수
sv_max	전체 서버 분포 중 계정이 접속한 서버들의 분포의 최대값
sv_min	전체 서버 분포 중 계정이 접속한 서버들의 분포의 최대값
sv_mean	전체 서버 분포 중 계정이 접속한 서버들의 분포의 평균값

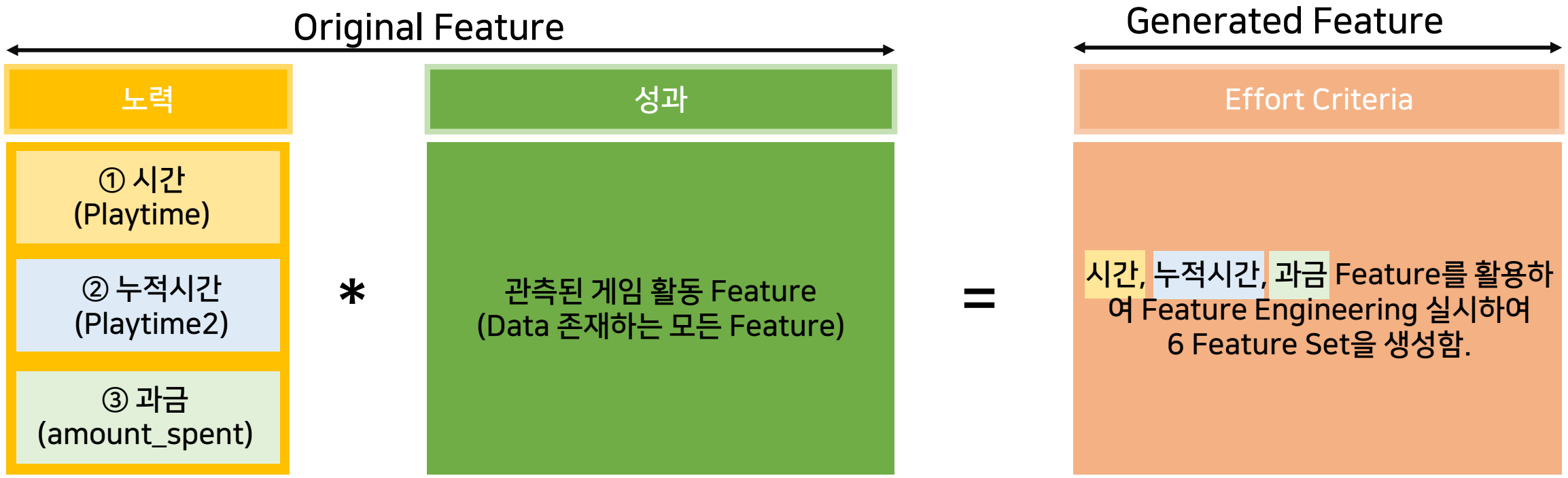
파생변수 생성 기준

- Our Assumption : 노력보다 성과가 안 나왔을 때 이탈할 것이다.

Case 1 : 매일 Play를 했으나 원하는 바를 이루지 못해 흥미를 잃어 이탈하였다.

Case 3 : 한달동안 Play를 했으나 원하는 바를 이루지 못해 흥미를 잃어 이탈하였다.

Case 3: 과금을 하였지만 아이템 강화에 실패하여 이탈하였다.



Effort Criteria Feature

- 다음과 같은 기준으로 6개의 Effort Criteria Feature Set을 생성함.

① 시간 (Playtime)

- 1) 시간 대비 Data(EC1_df): day별 playtime 투자 대비 성과가 좋지 않을 경우 이탈하는 특성을 반영한 데이터
- 2) 시간 가중 Data(EC2_df): day별 playtime을 많이 투자할수록 이탈하지 않는 특성을 반영한 데이터

② 누적시간 (Playtime2)

- 3) 누적시간 대비 Data(EC3_df): day별 누적 playtime을 많이 투자할수록 이탈하지 않는 특성을 반영한 데이터
- 4) 누적시간 가중 Data(EC4_df): day별 누적 playtime을 많이 투자할수록 이탈하지 않는 특성을 반영한 데이터

③ 과금 (amount_spent)

- 5) 과금 대비 Data(EC5_df): amount_spent 투자 대비 성과가 좋지 않을 경우 이탈하는 특성을 반영한 데이터
- 6) 과금 가중 Data(EC6_df): amount_spent 를 많이 투자할수록 이탈하지 않는 특성을 반영한 데이터

Effort Criteria Feature

- 6개의 파생 데이터 생성

ex) 과금 대비 Data 생성

2day1

day	acc_id	npc_kill	revive
1	75001	aa	1.441844
1	75001	aa	0.283219
…	…	…	…

X

2day1

amount_spent

2day2

day	acc_id	npc_kill	revive
1	75001	aa	1.441844
1	75001	aa	0.283219
…	…	…	…

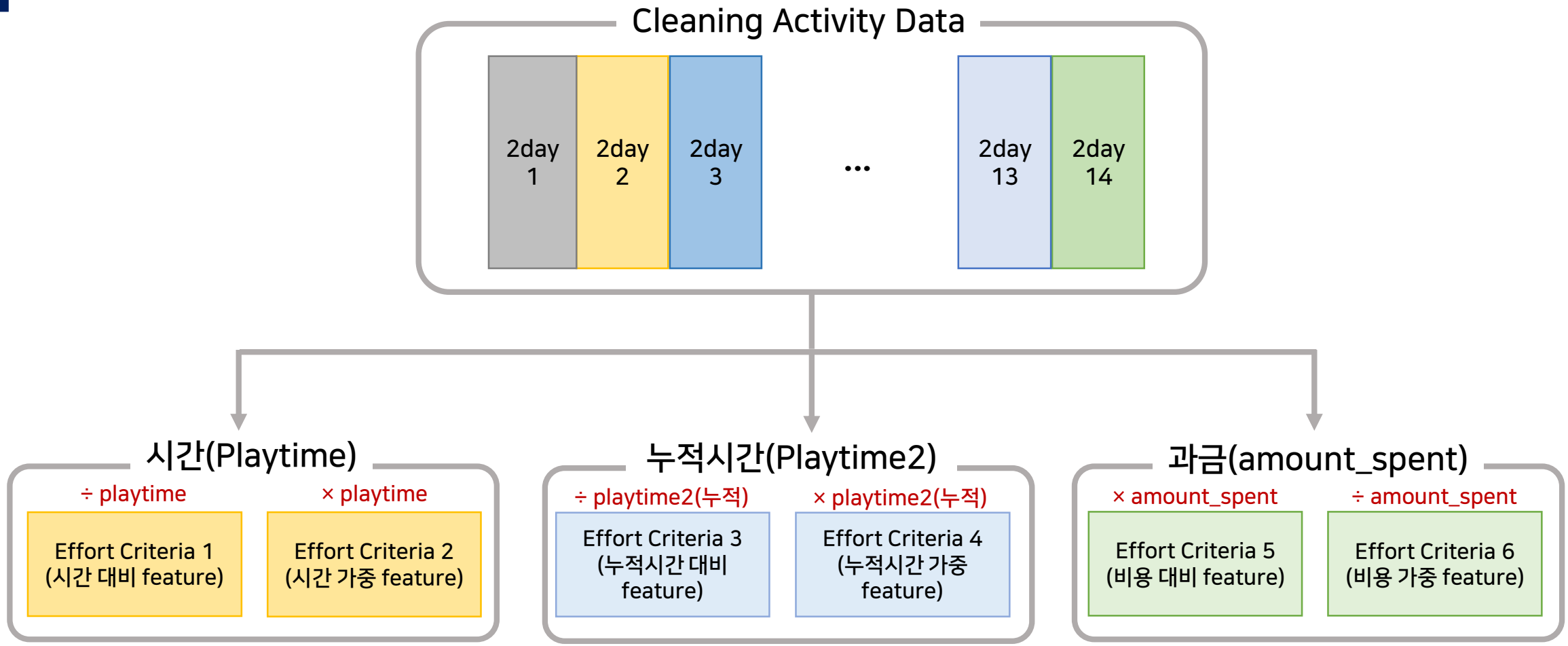
X

2day2

amount_spent

- 정제한 activity data에 각 2day에 해당하는 playtime, playtime2(누적), amount_spent변수 값을 곱하거나 나누는 연산을 통해 각각 다른 의미를 나타내는 6개의 파생 데이터를 생성함.

Effort Criteria Feature

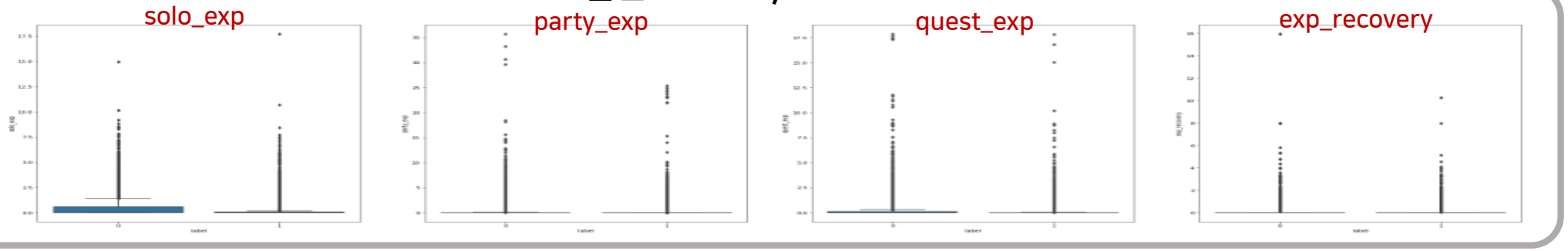


- Activity Data로부터 각각 다른 의미를 지닌 6개의 파생 데이터를 생성함

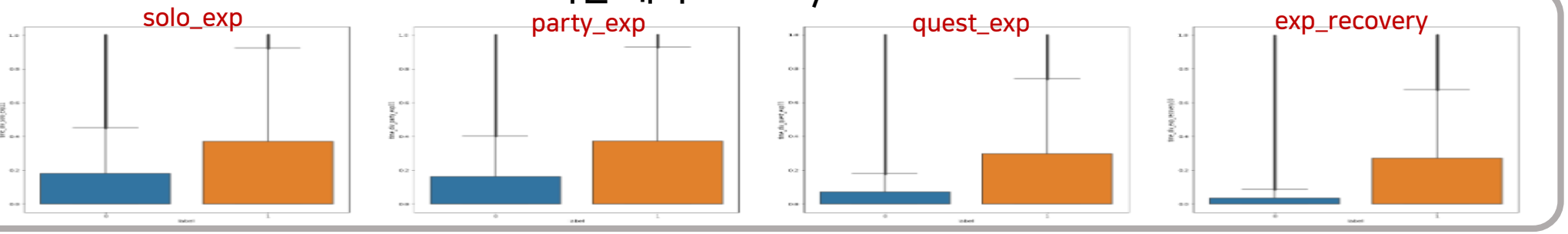
Effort Criteria Feature

- 생성된 Effort Criteria Feature가 유의미한 것인지 확인하기 위해 Visualization을 진행.
- 전체적으로 잔존 유저와 이탈유저의 분포가 차이가 있으나 시간대비 Data의 시간 당 경험치 관련 Feature에 특히 큰 차이가 있는 것을 확인.

일반 Activity Feature



시간 대비 Activity Feature



- 시간으로 나눈 분포를 시각화한 결과, 기존에는 보이지 않았던 이탈, 잔존 유저 간의 분포차이가 있음을 확인.

Sub Features Data

Trade Data의 feature 생성

- 1) 거래 item : acc_id별 거래한 item에 대한 변수 생수 생성
  ex) 거래한 아이템 수량의 합, 거래한 아이템 가격의 합
- 2) 거래 ID : acc_id별 보유 케릭터의 Level에 대한 변수 생성
  ex) 거래(trade) 횟수, 자기 계정 내 거래 횟수의 합

PvP Data의 feature 생성

- 1) Level : 보유 케릭터에 대한 Level관련 변수 생성
  ex) 보유 케릭터 레벨 관련 변수, 보유 케릭터 레벨업 관련 변수
- 2) 전투활동 : 캐릭터 전투활동 관련 변수 생성
  ex) 총 혈맹 전투 참여 횟수, 총 무작위 공격을 행한/받은 전투 횟수, 단발성 전투의 합

Pledge Data의 feature 생성

- 1) 혈맹이 속한 ID의 접속기록 : 해당 아이디가 속한 혈맹에 접속한 캐릭터관련 변수 생성

ex) 혈맹의 수, acc_id 별 혈맹에 접속한 캐릭터 수의 통계치

- 2) 전투기록 : 혈맹 Battle과 관련된 변수 생성

ex) 혈맹 전투 횟수의 통계치, 혈맹이 막피공격을 행한(받은) 전투 횟수에 대한 통계치

Difference Data의 feature 생성

- 1) 시간차이 : 누적 시간 대비 / 가중치관련 변수 생성

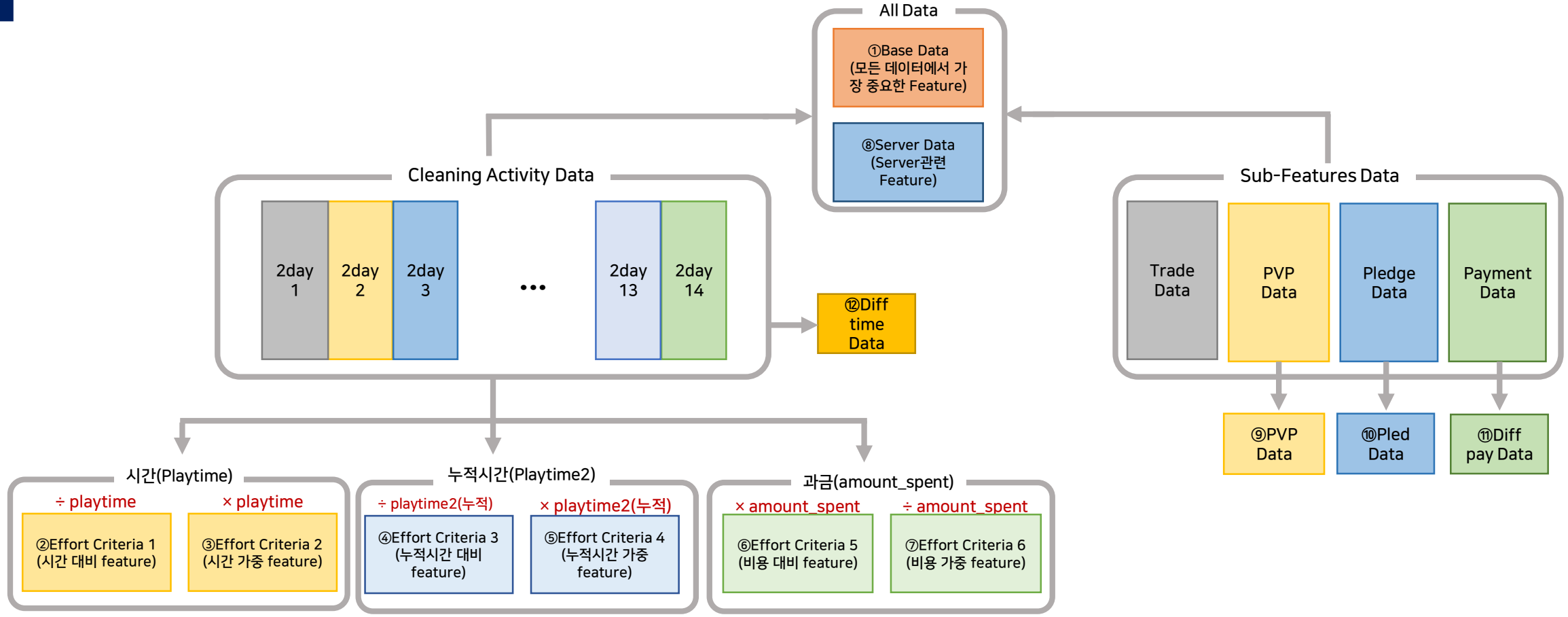
ex) 누적 playtime로 나누어준 변수들의 (마지막 주차 - 1주차) 변수들

- 2) 과금차이 : 누적 과금 대비 / 가중치관련 변수 생성

ex) 누적 amount_spent로 나누어준 변수들의 (마지막 주차 - 1주차) 변수들

최종 데이터

Original Feature : 73 → Generated Feature : 1109

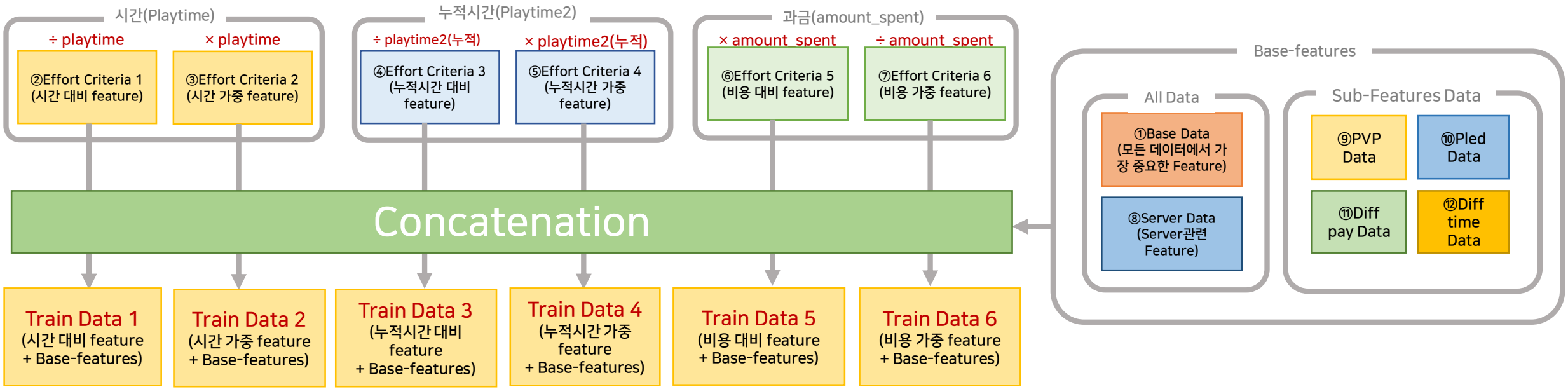


- Activity data와 Payment data를 통해 생성한 6가지 파생 데이터에 Sub-features Data로 정의한 정제된 Trade data, PvP data, Pledge data, Pavement data를 통해 최종 데이터 12 Data Set을 생성함.
- Feature의 개수 경우 73개에서 1109로 변화하였고 추후에 12개의 Data set을 고려하여 모델링을 진행.

AGENDA

1. 데이터 전처리
2. 모델링
3. SHAP Value
4. 결과 및 해석

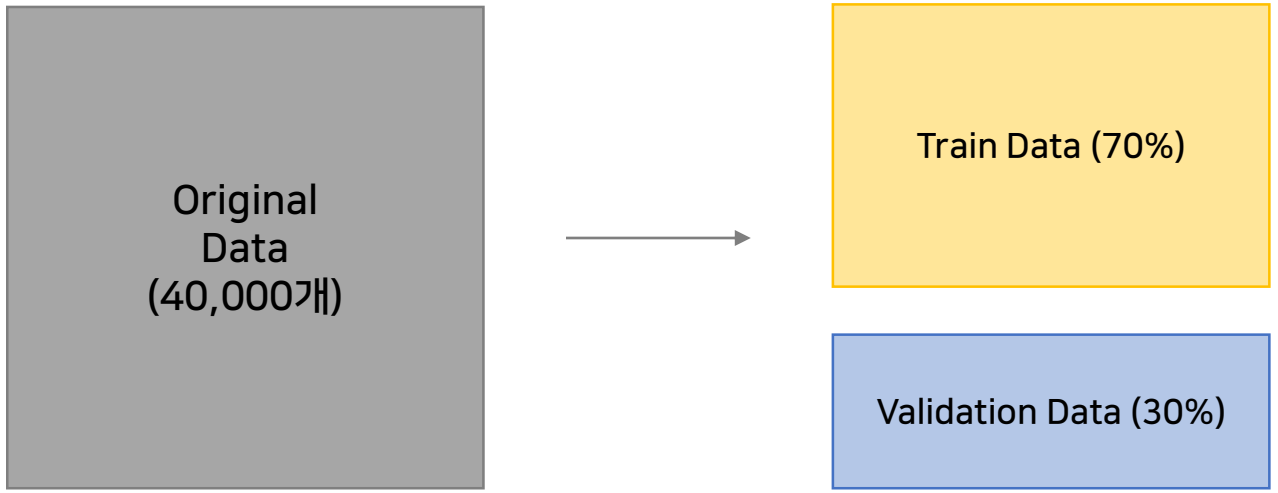
Train Data Making



- Activity data와 Payment data를 통해 생성한 6가지 파생 데이터에 Base-Features로 정의한 정제된 Base Data, PvP Data, Pledge Data, Diff Data, Server Data를 concatenate해서 학습 데이터 6개를 생성함.

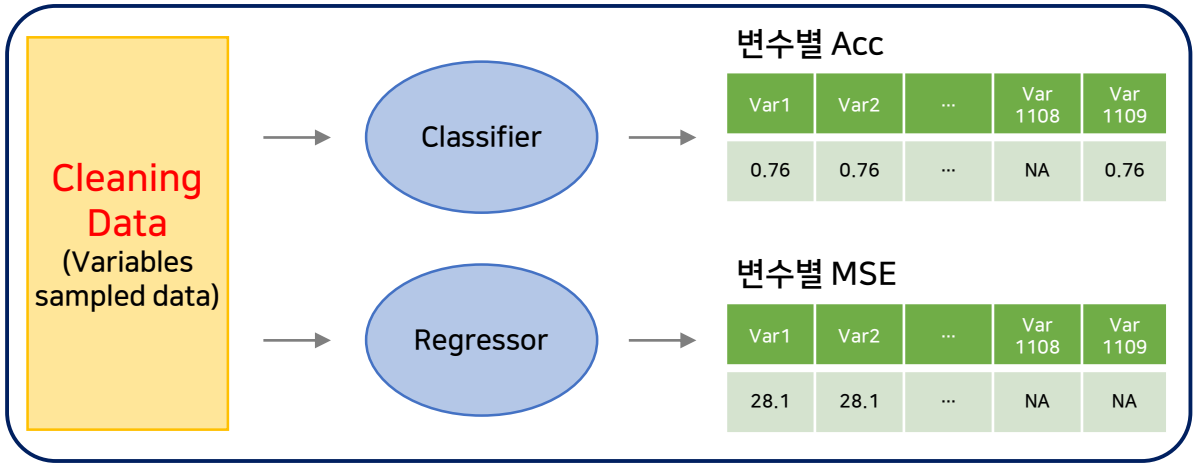
Data Split

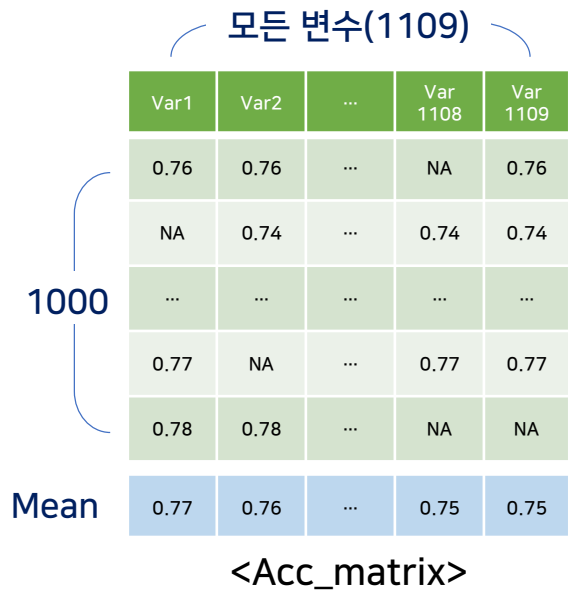
- Data Split Rule



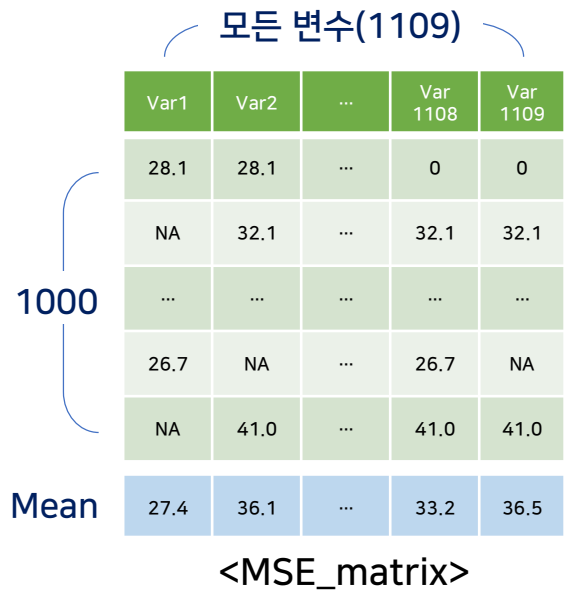
- 전체 데이터를 train : validation 데이터의 비율이 7:3이 되도록 분할함
- Train data로 학습한 모델을 validation data에 test하며 최적의 parameter를 탐색함

Feature Selection





<Acc_matrix>

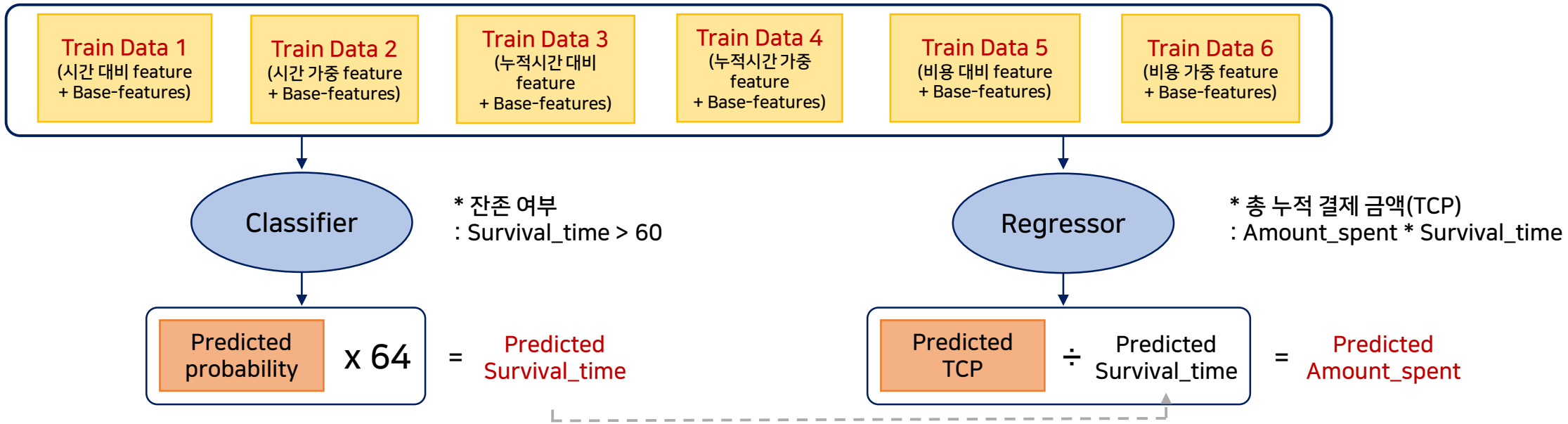


<MSE_matrix>

- Step 1) 총 변수를 묶어서 25% 변수 샘플링한 데이터로 모델링 진행
- Step 2) 선택된 변수에 Acc, MSE 성능 기록
- Step 3) 앞선 단계 1000번 수행 후 상위 75% 변수 사용

모델링 구조

모델 concept



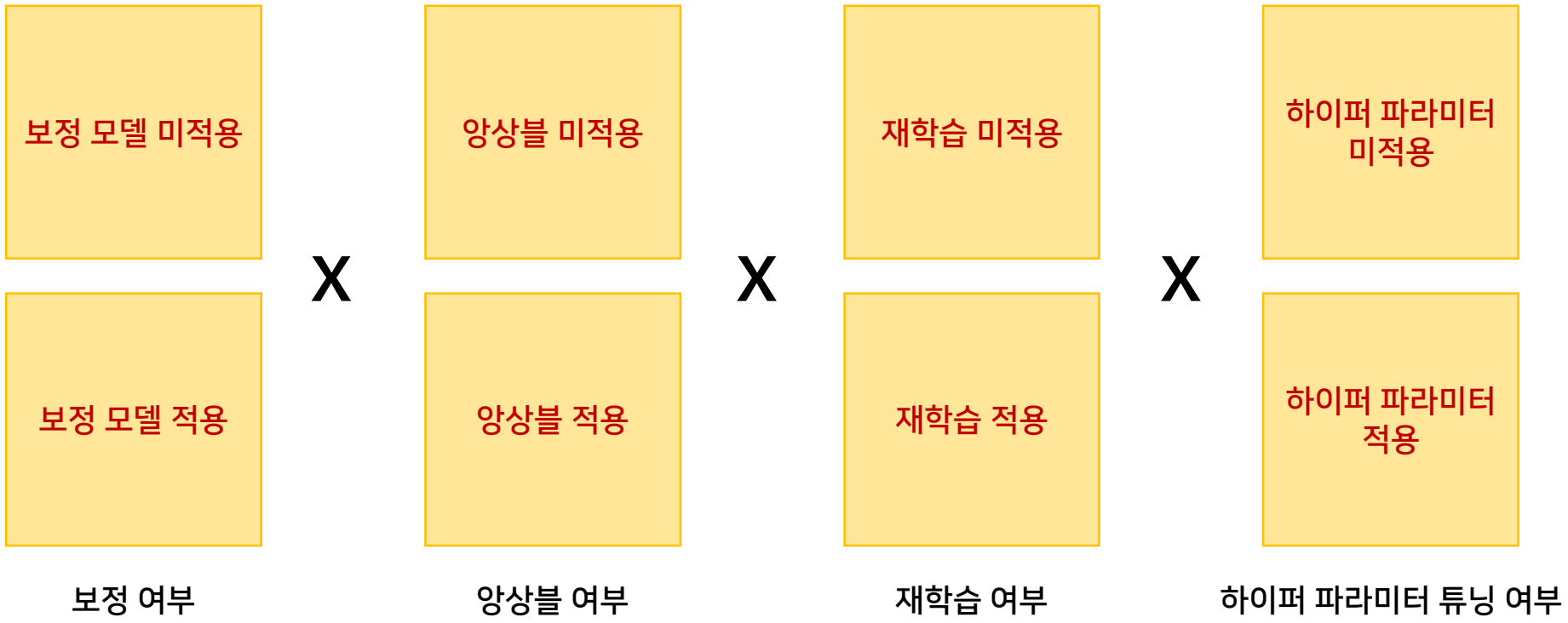
- Step 1) 잔존 여부를 예측하는 classifier 구축
- Step 2) 0~1 사이값에 64를 곱하여 survival_time form 도출

- Step 1) 총 누적 결제 금액(TCP)를 예측하는 Regressor 구축
- Step 2) 앞서 예측된 잔존 일수로 나누어 일별 평균 결제 금액 예측

Regressor의 성능을 Classifier에 종속시킴으로써 생존 기간 예측률 개선을 통해 전체 성능 향상을 도모

모델 극대화 전략

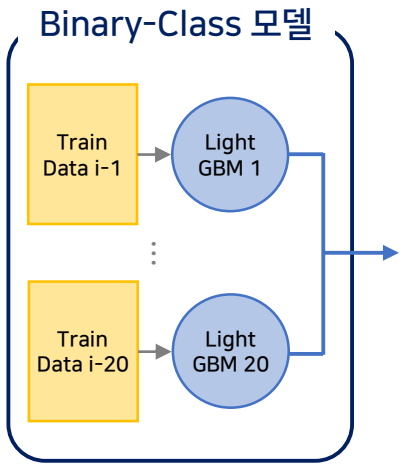
- 모델 극대화 전략 Description



- Regressor 모델 보정, 앙상블, 재학습, Bayesian opt.를 각각 A/B Test 진행
- Validation set 검증 결과를 바탕으로 최적의 조합 도출

모델 극대화 전략

① Regressor 보정 모델

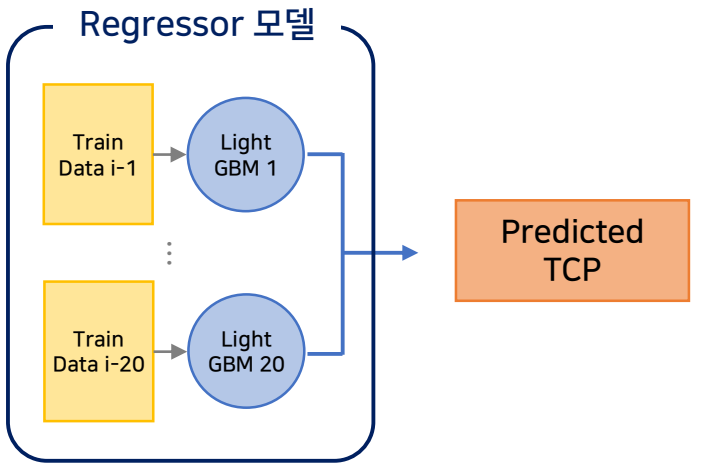


- 목적 : 과금 유저와 무과금 유저의 분류
- Label : Amount_spent가 0 초과시 1, 아니면 0
- 예측 확률이 5%미만인 경우 무과금 유저로 판단

<Ex>

Acc_id	P(과금유저)	P(무과금유저)
1	0.03	0.97

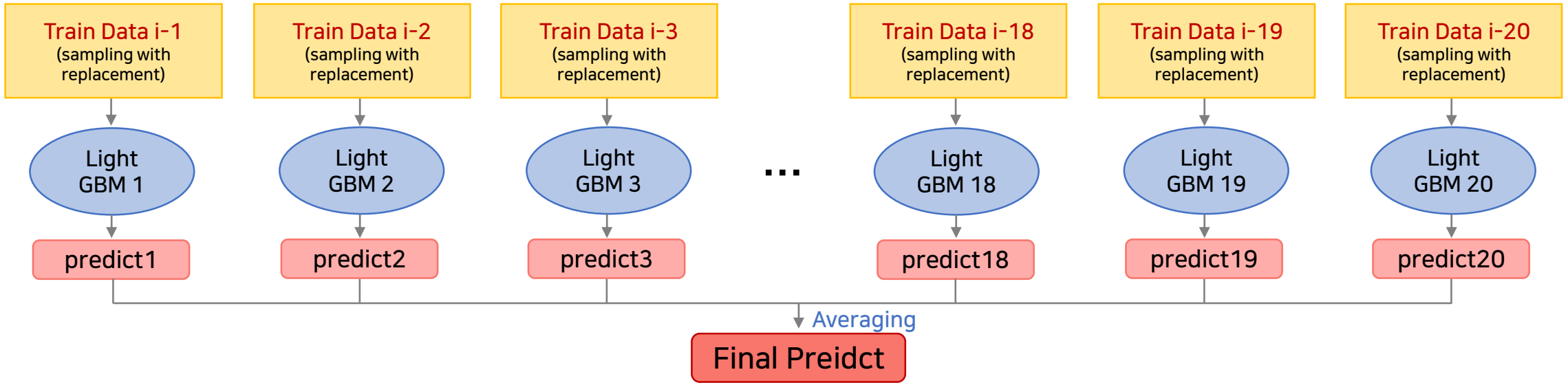
=>1번 계정의 TCP를 0으로 할당 후 Regressor 학습



- 전체 중 무과금 유저(Amount spent=0)는 16,438명으로 전체의 41%에 해당
- Validation set에 Binary classifier 검증 결과 95%의 정확도를 보임
- Regressor 보정 모델을 적용한 결과 리더보드 기준 Score 약 300의 성능 향상

모델 극대화 전략

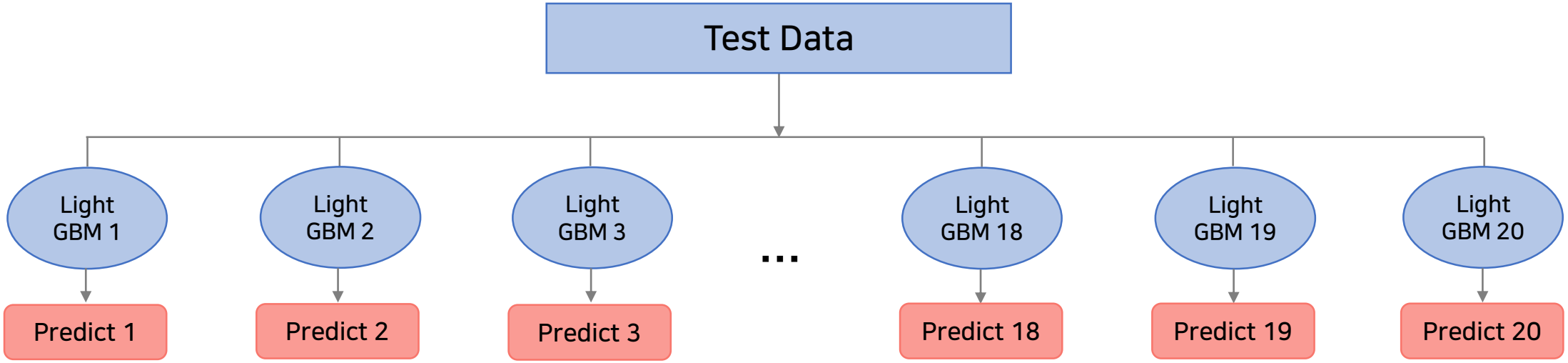
- ② 앙상블 : LightGBM Based Bagging



- Step 1) i번째 train data를 복원 추출 후 Light GBM 모델을 학습함
- Step 2) step 1을 20회 반복하여 총 20개의 Light GBM 단일 모델을 구축함
- Step 3) 20개의 Light GBM 모델의 averaging을 통해 최종 예측값을 도출
- 앙상블을 통해 구축된 모델은 단일 모델에 비해 리더보드 기준 Score 약 1,000의 향상을 보임

모델 극대화 전략

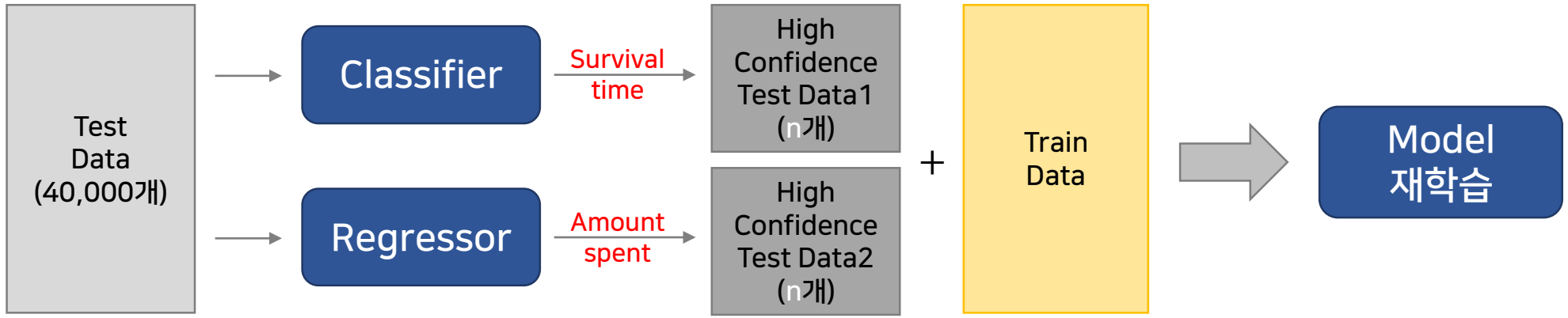
- ③ 재학습 : Confidence가 높은 Test data를 Train data에 추가



- 단일 모델 20개가 같은 predict값으로 예측한 test data는 학습 데이터로 재사용
- 1) Classifier : 예측값의 표준편차가 0.03이내인 경우
- 2) Regressor : 예측값의 표준편차가 1.5이내인 경우

모델 극대화 전략

- ③ 재학습 : Confidence가 높은 Test data를 Train data에 추가

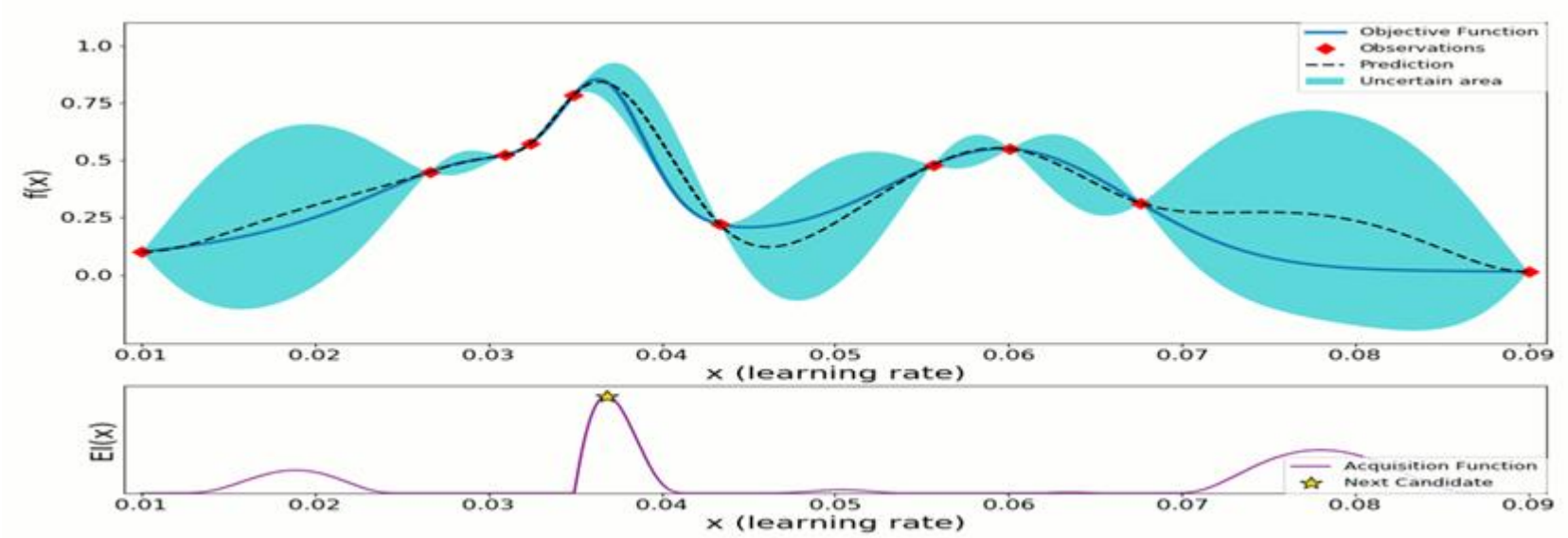


- 20개 단일 모델의 예측 값이 임계 표준편차 이내인 predict값의 평균을 true value로 가정
- Train data에 위의 데이터를 추가하여 기존 모델을 재학습함
- 데이터 개수의 증가를 통해 리더보드 기준 Score 약 300의 향상을 보임

모델 극대화 전략

- ④ 하이퍼 파라미터 튜닝 : Bayesian optimization

*출처 :Sualab Research Blog – Bayesian Optimization 개요



- LightGBM의 하이퍼 파라미터 튜닝 시 목적함수를 MSE나 Acc가 아닌 Score_function으로 설정
- Bayesian Optimization을 통해 불필요한 탐색을 최소화하여 효율적인 Tunning 진행
- 파라미터 튜닝 모델은 리더보드 기준 Score 약 400의 향상을 보임

모델 A/B Test 결과

Data	앙상블 여부	Base	Base+보정	Base+튜닝	Base+보정+튜닝	Base+재학습+튜닝	Base+보정+재학습+튜닝
Activity data1	단일모델	6332.343258	6543.724312	6515.23241	6705.059262	7140.387012	7600.48512
	앙상블	6854.187616	6941.565504	7016.539694	7017.525712	7352.115502	7535.712412
Activity data2	단일모델	7195.37442	7236.459036	7130.975668	7548.407632	6788.172356	7265.138576
	앙상블	7345.69571	7531.911768	7329.849302	7222.357642	7038.468688	7332.42045
Activity data3	단일모델	7063.179288	7261.755898	7158.25768	7377.564082	6602.674412	6910.829288
	앙상블	7698.723536	7513.002168	7526.749946	7620.689454	7025.740758	7261.480396
Activity data4	단일모델	6532.701032	6747.958992	6844.985232	7088.341638	6624.850892	6872.196714
	앙상블	7339.278504	7412.216716	6995.518976	7069.215944	6732.588328	7255.411032
Activity data5	단일모델	6164.967492	6414.369482	6055.706094	6337.41995	7416.745168	7823.079448
	앙상블	7113.699626	7286.144502	7221.250822	7415.433158	7706.581578	8252.203606
Activity data6	단일모델	7014.669166	7243.461542	7021.686006	7254.642598	6840.61166	7174.48115
	앙상블	7746.719064	7583.059106	7484.362202	7589.35485	7591.82096	7963.008702

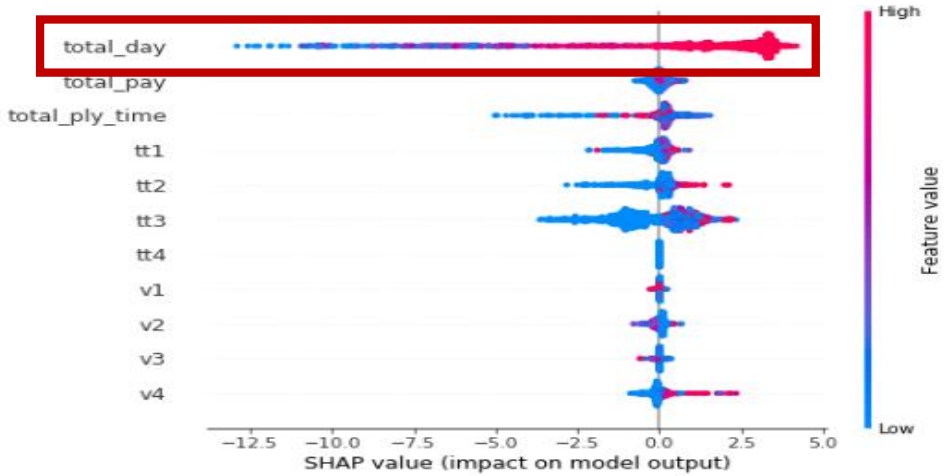
Validation 셋에 적용 결과 앙상블 + 보정 + 재학습 + 하이퍼 파라미터 튜닝을
전부 적용하였을 때 가장 좋은 성능을 보이며 해당 모델로 리더보드 기준 2위 달성

AGENDA

1. 데이터 전처리
2. 1차 모델링
3. SHAP Value
4. 결과 및 해석

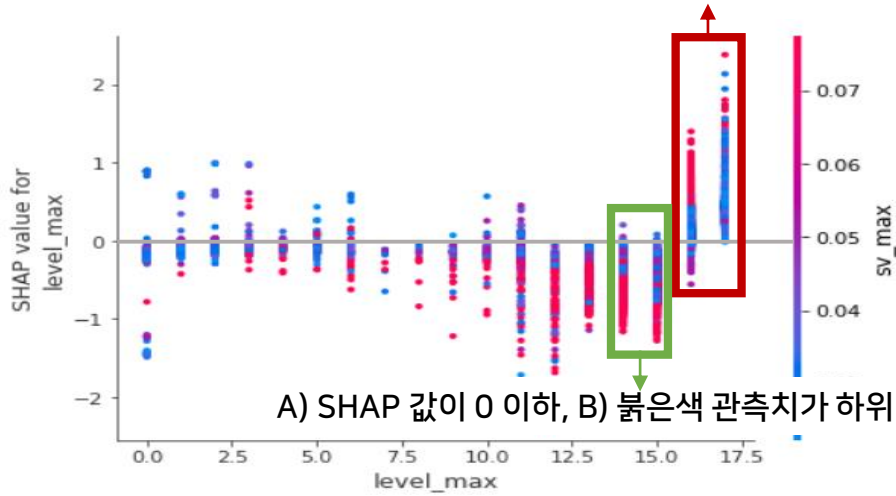
SHAP란?

- SHAP values : 각 feature가 model의 output에 미치는 영향의 정도를 나타냄
  - Classifier와 Regressor 모델 각각에 대해 SHAP를 적용 후 변수 해석 진행
- 잔존 여부 Classifier 모델 SHAP 예시



- 여러 개의 feature별로 모델의 output에 미치는 영향을 확인
- SHAP value(x축) 0 이상이면 영향력이 큼, 미만은 작음
- 붉은색일 수록 해당 feature의 값이 높음
- "total_day" 해석 : 접속 일수가 많은 유저일 수록 잔존 확률이 높다

A) SHAP 값이 0 이상, B) 붉은색 관측치가 상위 분포

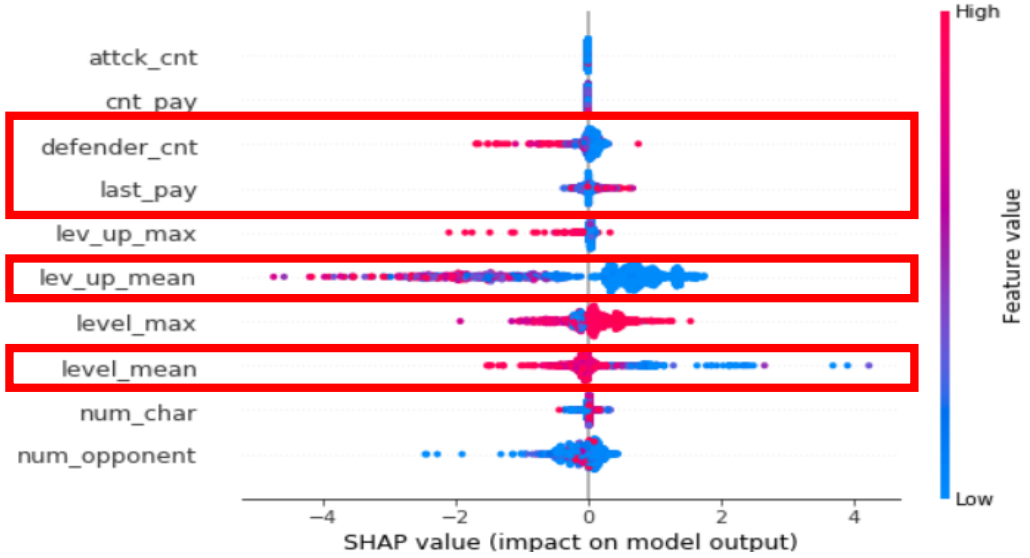


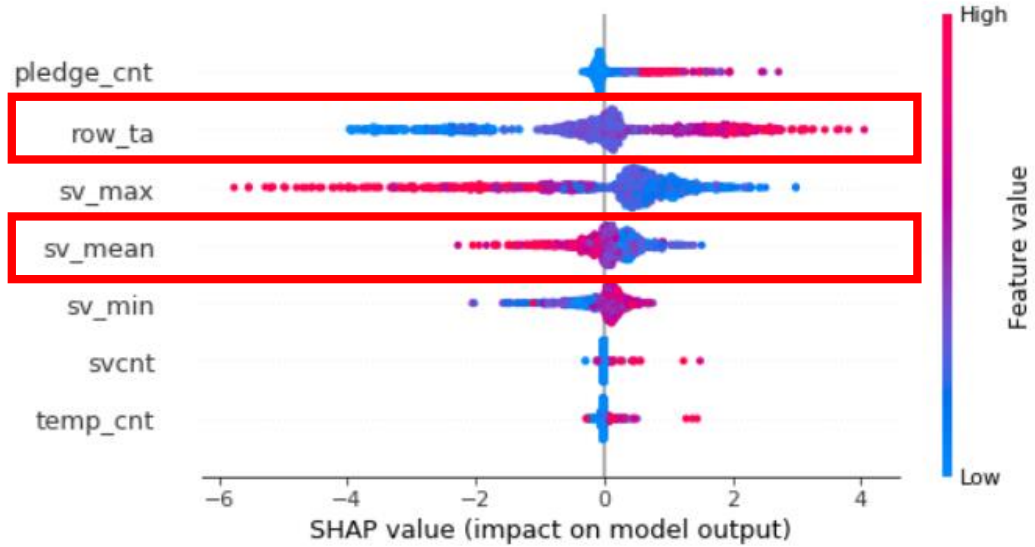
A) SHAP 값이 0 이하, B) 붉은색 관측치가 하위 분포

- 특정 feature A, B가 모델의 output에 미치는 영향을 확인
- 붉은색일 수록 feature B(sv_max : 높으면 기존 서버)의 값이 높음
- 해석 : A)고 레벨일 경우 잔존 확률이 높으나 특정 고 레벨 이전엔 이탈할 확률이 높으며 B)그 현상은 기존 서버에서 두드러지게 나타남

1. 잔존 여부 Classifier 모델 SHAP Value

- Base Feature에 대한 SHAP Value 해석

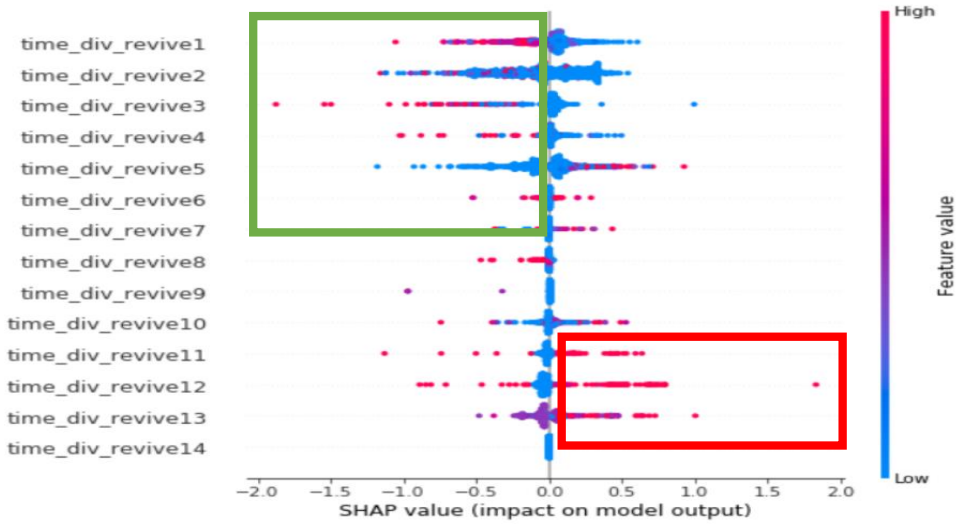




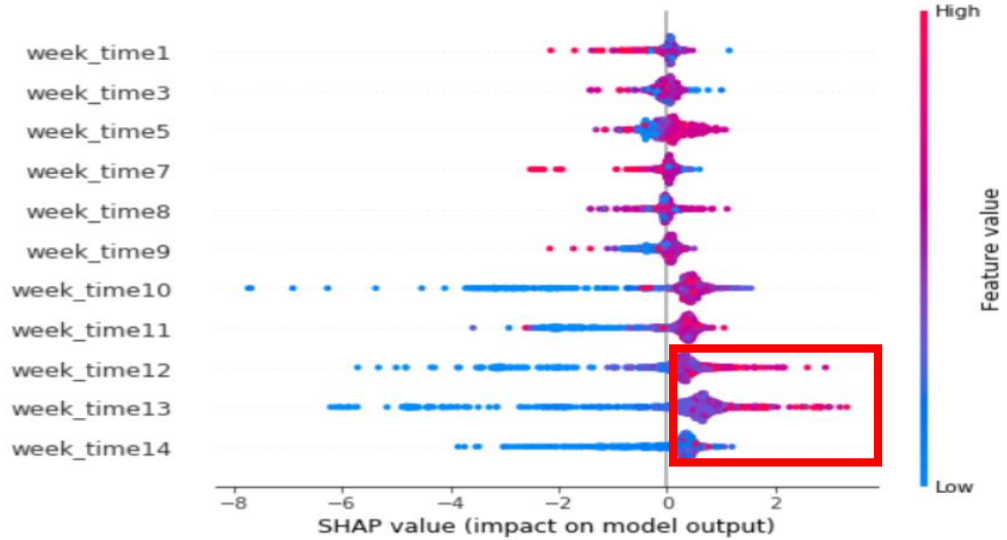
Target Feature	Description
defender_cnt	무작위 공격(막피)의 피해량이 많을수록 이탈 확률이 높음
last_pay	마지막 일자에 과금을 한 유저는 잔존할 확률이 높음.
lev_up_mean	단기간내에 레벨업이 많을 수록 이탈 확률이 높음, 꾸준히 하는 유저가 이탈 확률이 적음.
level_mean	평균 레벨이 낮은 유저는 잔존할 확률이 높음.
row_ta	활동량이 많은 유저는 잔존할 확률이 높음.
sv_mean	신생서버일수록 잔존할 확률이 높음.

1. 잔존 여부 Classifier 모델 SHAP Value

- 플레이 시간대에 따른 SHAP Value 해석



(a)

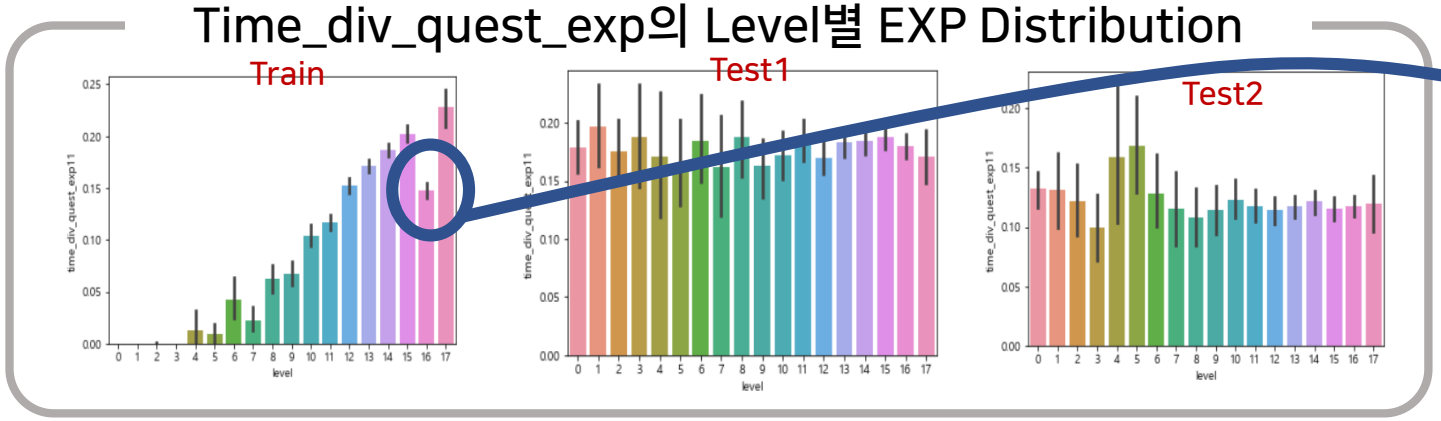


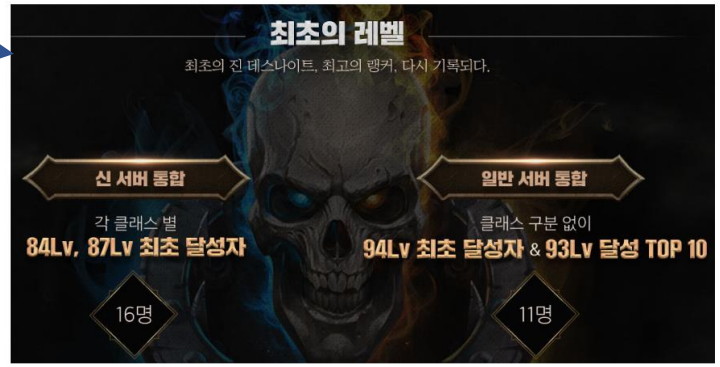
(b)

- 부활 횟수(a) : 28일 중 초반에 부활 횟수가 높은 유저는 이탈할 가능성이 높으며 후반으로 갈수록 부활 횟수가 높은 유저는 잔존 확률이 높아짐
- 2일당 플레이 시간 (b) : 초반에는 크게 두드러지지 않으나 후반부에 많이 플레이하는 user가 잔존 확률이 높음

2. 총 누적 과금 금액 Regressor 모델 SHAP Value

- Pattern① : High Level User일 수록 과금을 많이 하는 성격이 미래에 유지되는 경향성이 존재함.
- 레벨이 높을 수록 획득 경험치가 높을 것이라는 가정으로 시각화를 진행하였고 이를 확인하여 3개의 High Level User Feature를 설정하였음.
- High Level User Feature : max_level(계정 별 캐릭터 중 고랩) / time_div_solo_exp(시간대비 솔로 경험치) / time_div_npc_kill(시간대비 NPC를 죽인 횟수) / time_div_quest_exp(시간대비 퀘스트 경험치)

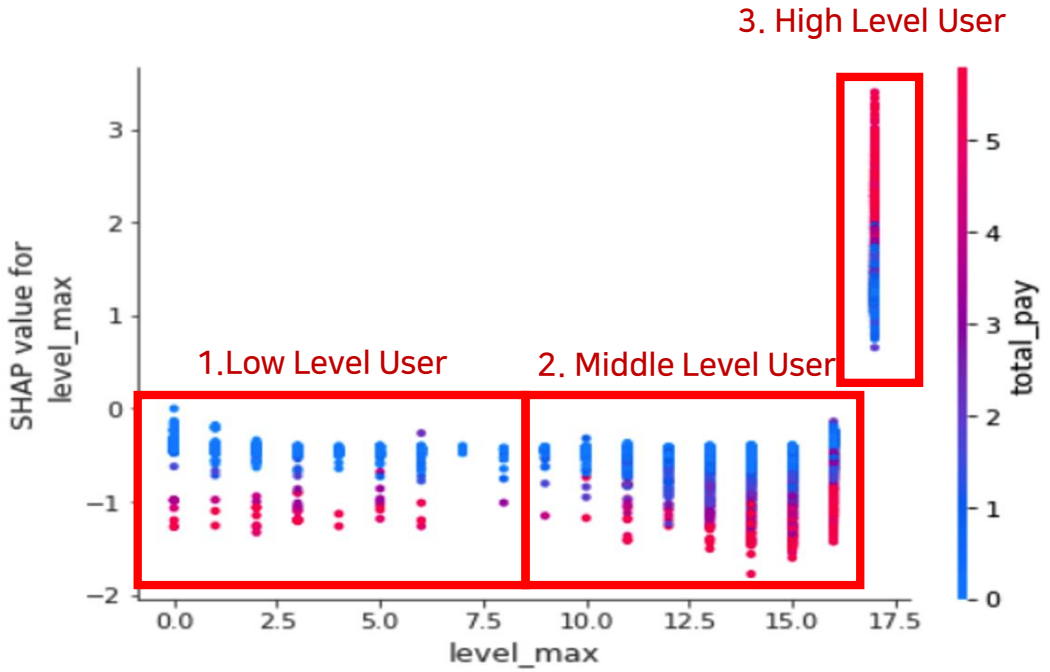




- Train Data기간에 레벨이 클수록 퀘스트 경험치가 증가하는 추세와 16레벨의 퀘스트 경험치가 떨어지는 것으로 보아 위와 같은 이벤트를 통해 높은 레벨 유저들이 17레벨의 달성을 위한 현상이 일어나는 것으로 보임.
- 후에 Test1에서는 이벤트 종료 후 이러한 경향성이 사라지고 Test2에서는 4,5레벨의 경험치 증가로 보아 신규 유저가 유입된 것으로 보임.

2. 총 누적 과금 금액 Regressor 모델 SHAP Value

- Pattern① : High Level User일 수록 미래에 과금을 많이 하는 성격이 유지되는 경향성이 존재함.

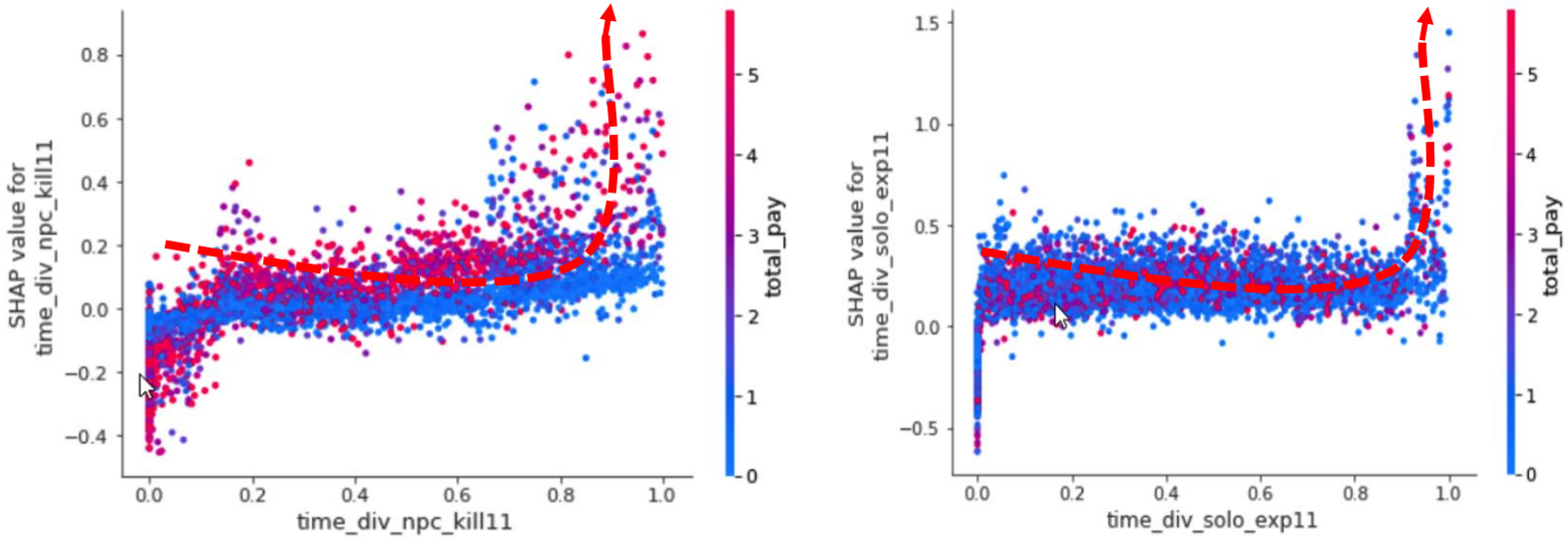


(a)

- 1) High Level User인 경우 빨간색 점이 위로 형성 : Train 구간에서 과금을 많이 하면 28일 뒤에서 그 경향성이 유지됨.
- 2) Middle Level User인 경우 중립적 : Train 구간의 과금 성향이 반드시 미래 시점에 일반화 되지 않음.
- 3) Low Level User인 경우 파란색 점이 위로 형성 : Train 구간에서 과금을 많이 하면 미래 시점에서 그 경향성이 유지되지 않음.

2. 과금

- Pattern① : High Level User일 수록 미래에 과금을 많이 하는 성격이 유지되는 경향성이 존재함.

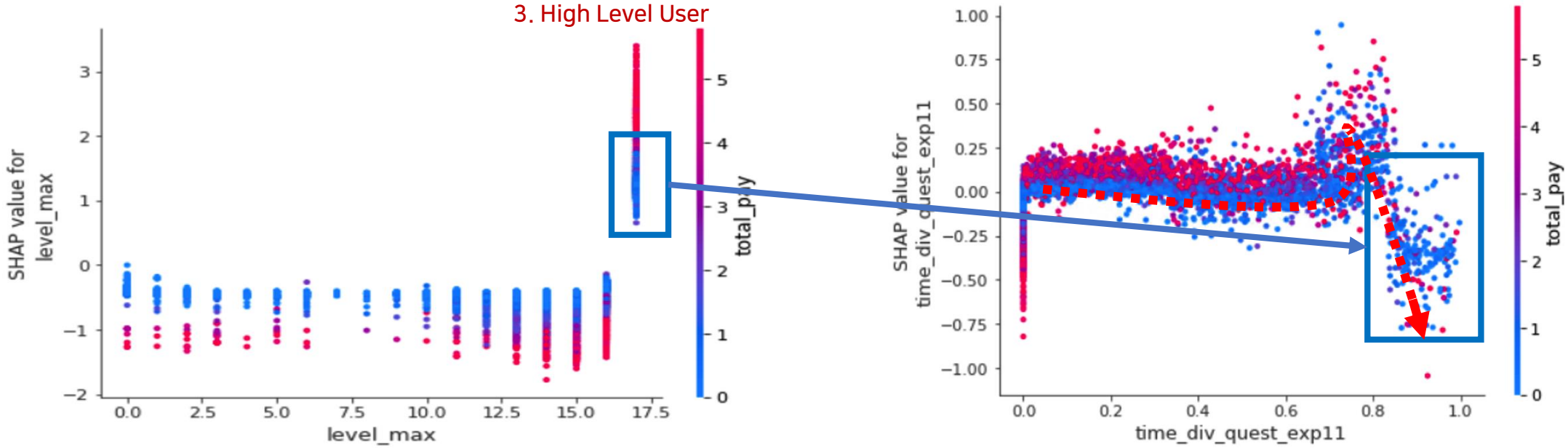


(b) (c)

- (b),(c) : EXP가 증가할 수록(=레벨이 높아질 수록) 빨간색 점이 위로 형성 : 레벨이 높아질 수록 Train 구간에서 과금을 많이 하면 Test 구간에서 그 경향성이 강화됨.

2. 총 누적 과금 금액 Regressor 모델 SHAP Value

- Pattern① : High Level User일 수록 미래에 과금을 많이 하는 성격이 유지되는 경향성이 존재하나, 퀘스트 경험치가 높은 User의 경우 현재, 미래 모두 과금을 하지 않는 현상이 존재.



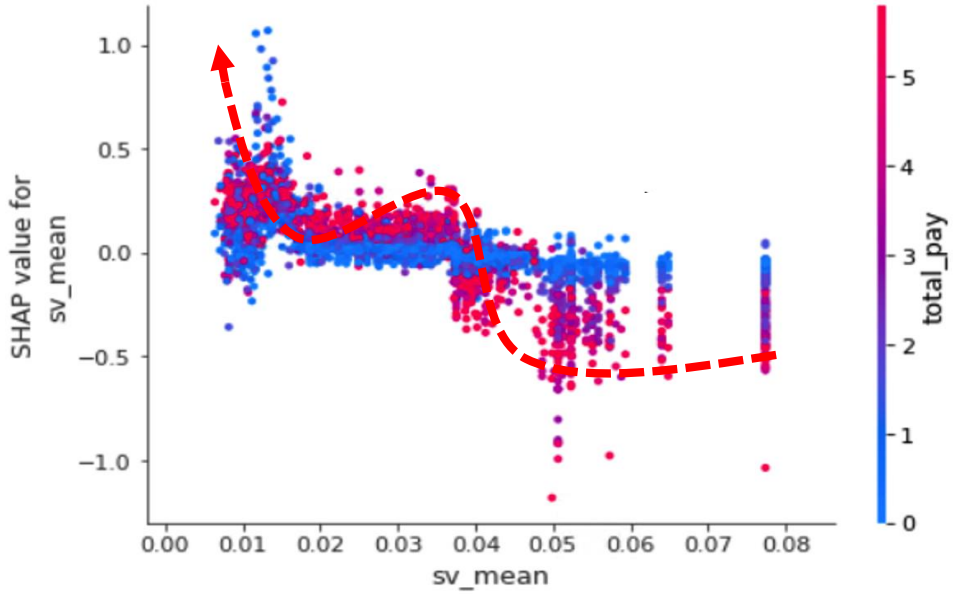
(a) (d)

- 1) Quest Exp가 증가할 수록 빨간색 점이 위로 분포 : 앞에서와 같이 퀘스트 경험치가 높은 유저(High Level User)일 수록 미래에 과금을 지속하는 경향성이 존재.
- 2) 특정 기점을 기준으로 빨간색이 아래로 분포하며 파란색 점이 다수 분포 : 퀘스트 경험치가 극단적으로 높은 유저(Highest Level User)일수록 현재, 미래 모두 과금을 하지 않는 경향성이 존재.

2. 총 누적 과금 금액 Regressor 모델 SHAP Value

- Pattern② : 신생서버일 수록 과금을 많이 하는 성격이 미래에 유지되는 경향성이 존재함.
- sv_mean이 낮을 수록 신서버를 의미하고 높을 수록 구서버를 의미함.

SHAP Value
28일 이후 과금액



Total_pay
학습 구간 내의 과금액

(e)

- 1.sv_mean이 작아질 수록 빨간색 점이 위로 형성 : 신서버일수록 과금을 많이 하는 성격이 미래에도(28일 뒤) 유지되는 경향성이 존재.
- 2.sv_mean이 작아질 수록 파란색 점이 위로 형성 : 구서버일수록 미래에 과금을 많이 하는 성격이 유지 되지 않음.

AGENDA

1. 데이터 전처리
2. 1차 모델링
3. SHAP Value
4. 결과 및 해석

최종 결론

- 잔존 여부 Classifier 모델 SHAP value 해석 결과
  - 무작위 공격의 피해량이 많은 user일수록, 특히 초반에 부활 수치가 높을 수록 이탈 확률이 높음
  - 단기간 내에 레벨업 수치가 많은 user일수록 이탈 확률이 높은 경향을 보임
  - 고 레벨(85 이상) 캐릭터를 보유한 user가 잔존 확률이 높으며 특히 기존 서버에서 두드러짐
  - 28일의 데이터 안에서, 신생 서버일 수록 잔존할 확률이 높음

- 총 누적 과금 금액 Regressor 모델 SHAP value 해석 결과
  - 레벨에 따라 과금의 성향이 다르며 High level user일수록 과금을 유지하는 경향을 보임(학습구간 내에서는 레벨 이벤트에 따른 현상으로 보임)
  - 다만 이 구간에서 퀘스트 경험치가 특히 많은 유저는 학습구간과 미래에서도 과금을 안하는 경향이 있음
  - 접속 서버에 따라 과금의 성향이 다르며 신생 서버일 수록 과금량이 높은 특성을 보임

최종 결론

- 잔존가치 극대화를 위한 idea 제안
  - 무작위 공격의 피해량이 많으며 특히 초반에 부활 수치가 높은 유저(초기유저이자 전투에 미흡한 유저)는 이탈 고위험군 분류 및 조치가 필요함
  - 단기간 내에 레벨업이 많은 유저의 경우 피로도를 해결할 수 있는 조치를 취하여 이탈 방지
  - 레벨 달성 이벤트 시 단계별 보상으로 상위 레벨(60~79) 구간의 유저를 후킹할 수 있는 컨텐츠 제시
  - 일반적으로 High level 유저의 경우 시점에 관계 없이 과금을 많이 하는 경향성을 유지하므로 주기적인 과금을 유도
  - 초 고레벨만을 위한 컨텐츠 제공
  - 신규서버 유저의 경우 잔존 확률이 높고 미래에 과금을 하는 경향성을 유지하므로 핵심적으로 관리할 필요성多

감사합니다