잔존가치를 고려한 이탈 예측 모형

팀명:리니지.csv

김종인
박선주
박진수

차례

01 문제소개

정확도? 기대이익 최대화!

문제. 이탈 징후를 보이는 고객을 사전 선별 -> 인센티브 제공으로 잔존 유도.

But '평균 과금액 ' 계산하여 수식화 한 'Score function'은 과금을 하는 유저들에게만 초점을 맞춤

따라서, 이탈징후 -> 과금을 하며 이탈 징후를 보이는 고객을 사전선별할것. !

Target Variables

1) survival_time : 이탈하는 날짜.

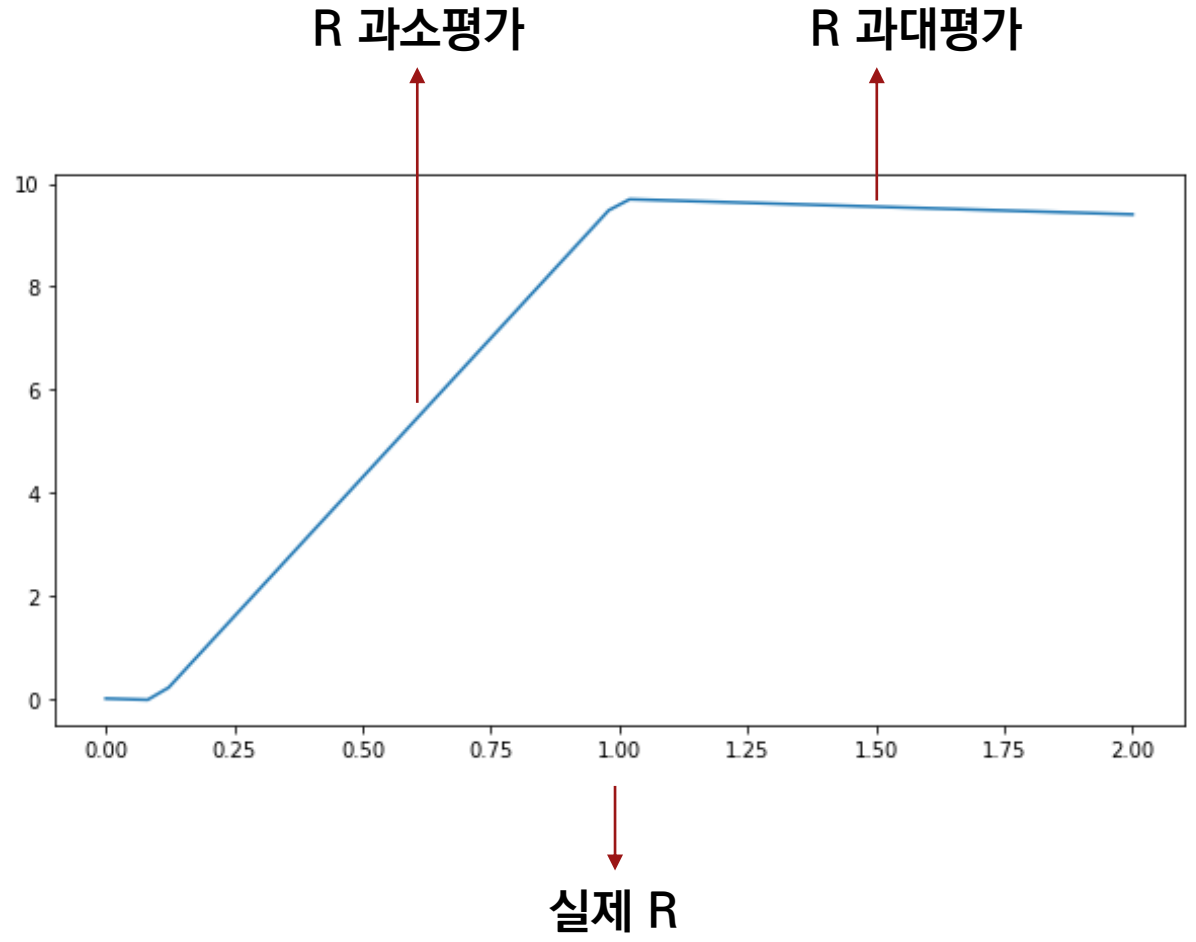
2) amount_spent : 평균 과금액.

01 분석의 목적 정확도? 기대이익 최대화!

- ✓ 정확도가 높으면 높을수록 좋다.
- ✓ 하지만 정확도를 높이는 것에는 한계가 존재한다.

기대이익을 최대화하는 관점으로 이탈 방지 전략을 세워야 한다

01 분석의 목적 Score Function 이해



과금액 예측의 경우 과소평가 보다는 과대평가하는 것이 낫다

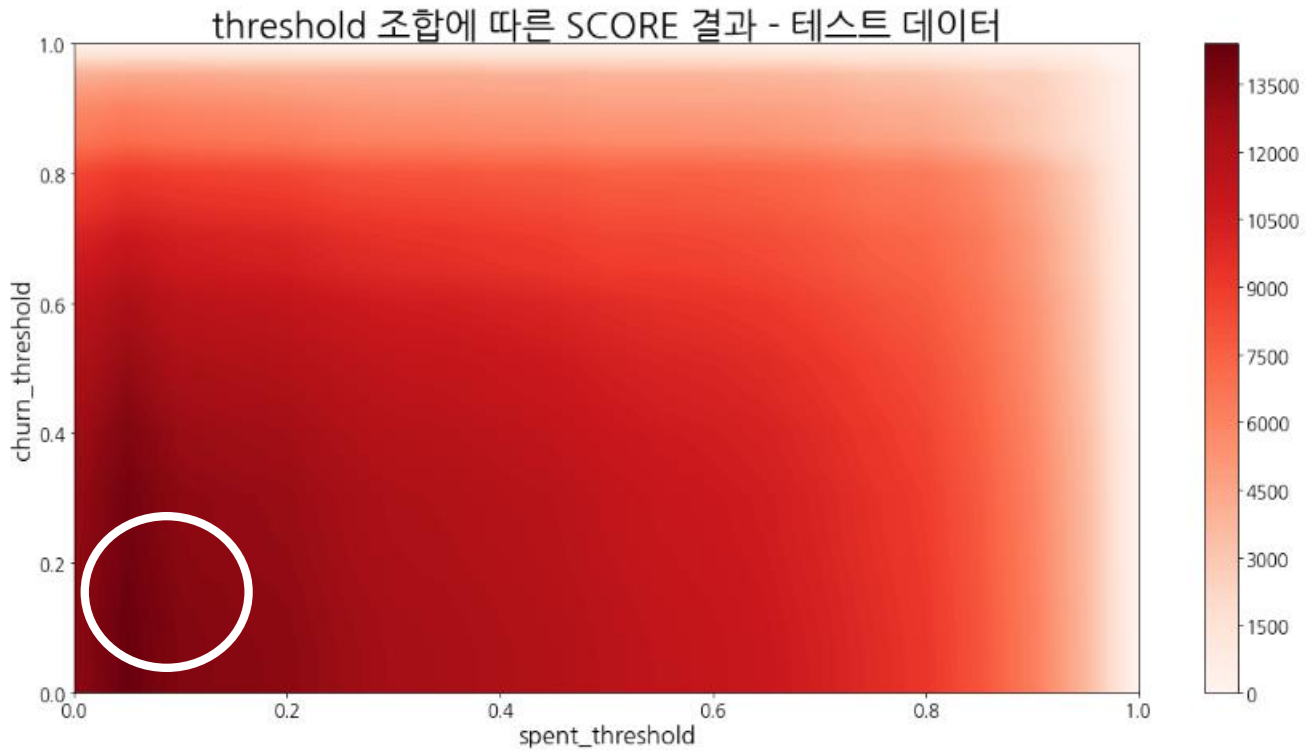
01 분석의 목적 Score Function 이해

Acc_id	생존기간 예측값	과금액 예측값	이탈 확률	과금 확률
A	23	3	0.45	0.43

분류모델에서 0.5 이하로 예측했더라도 이탈 방지 성공 시 보상 〉 비용

잔존확률 X 추가생존기간 X 일평균과금액 − 0.30 X 일평균과금액 〉 − 0.30 X 일평균과금액

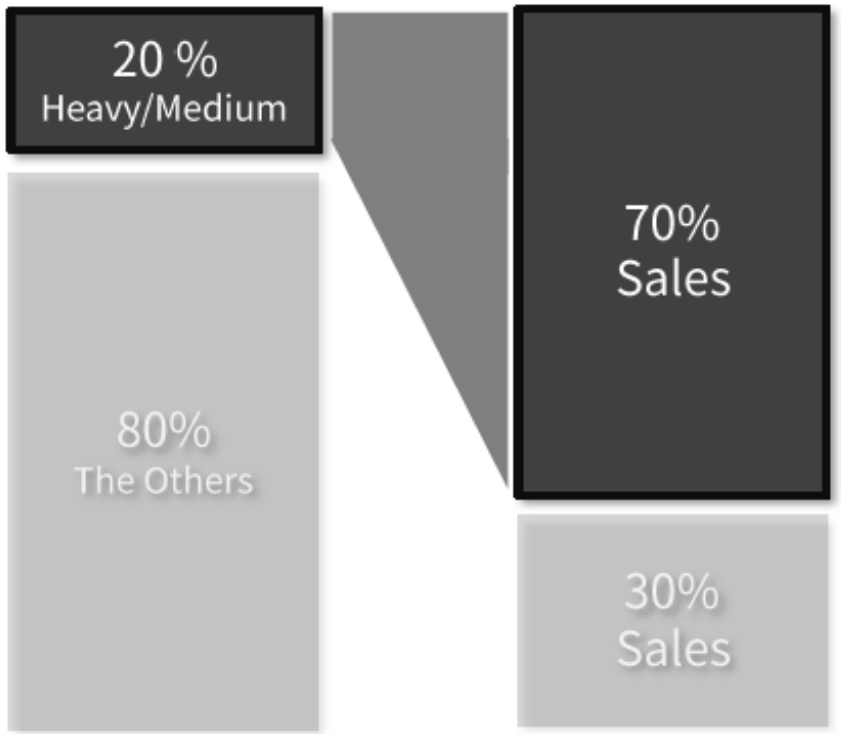
01 분석의 목적 Score Function 이해



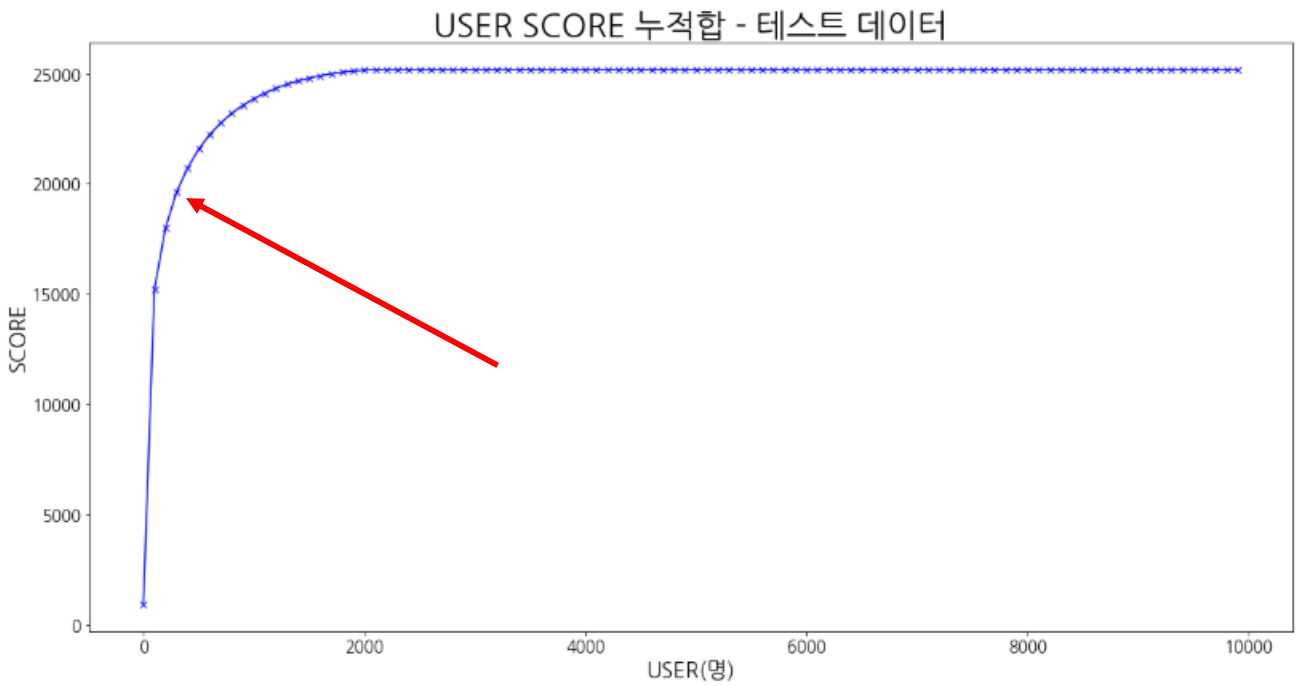
✓ 실제 기대 이익이 최대화 되는 지점은 threshold가 낮은 지점에서 형성된다. (train 기준)

✓ Threshold를 train 데이터를 바탕으로 하향 조정하는 전략

01 분석의 목적 기대이익을 위한 핵심유저!



20%의 고객이 70%의 매출 기여

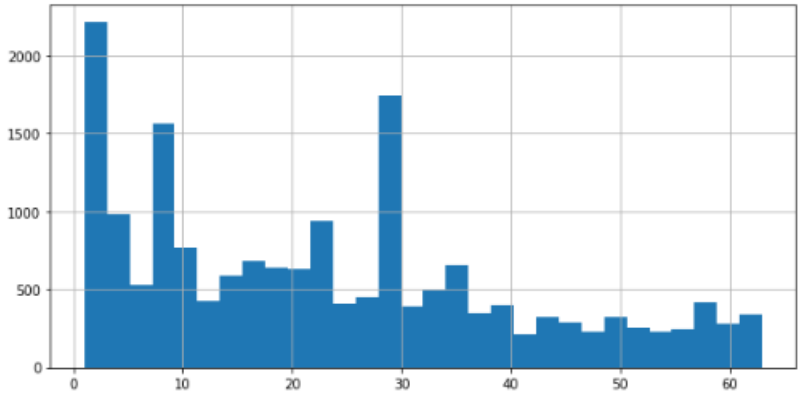


10,000명 중 **3%**인 300명의 이탈 방지를 성공하면

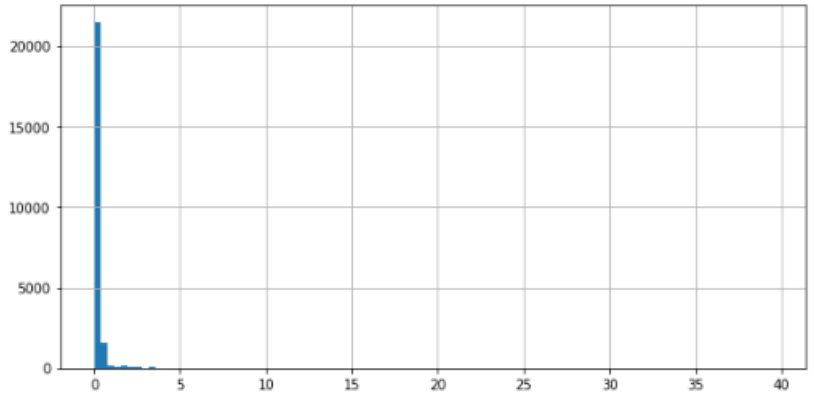
70%의 기대이익을 증가시킬 수 있다.

타겟 분포

Survival_time 히스토그램

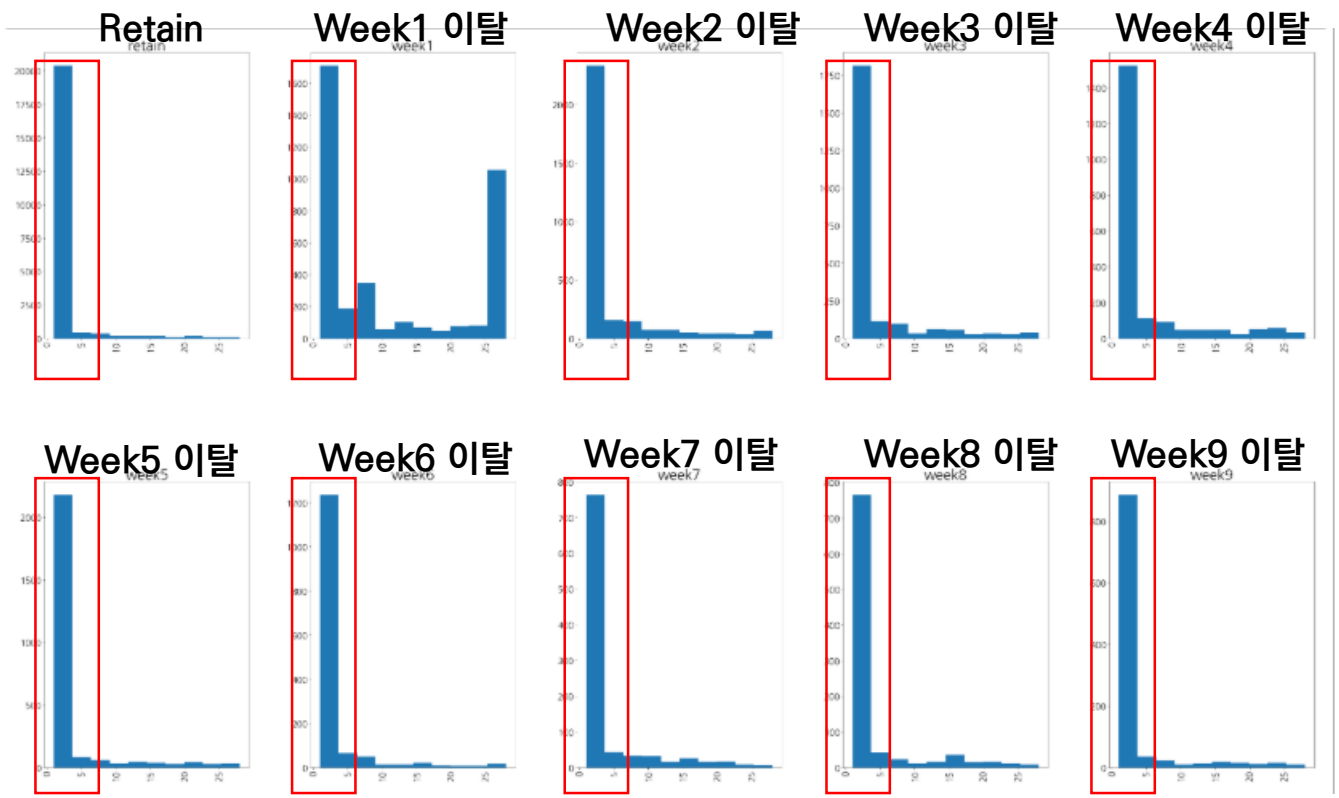


Amount_spent 히스토그램



- 잔존과 무과금 유저를 제외한 타겟의 히스토그램을 봤을 때 값이 커질수록 빈도수가 줄어드는 분포를 보인다.
- 일반적인 정규 분포의 성격을 띄지 않는다.

이탈 시기에 따른 최초 접속일



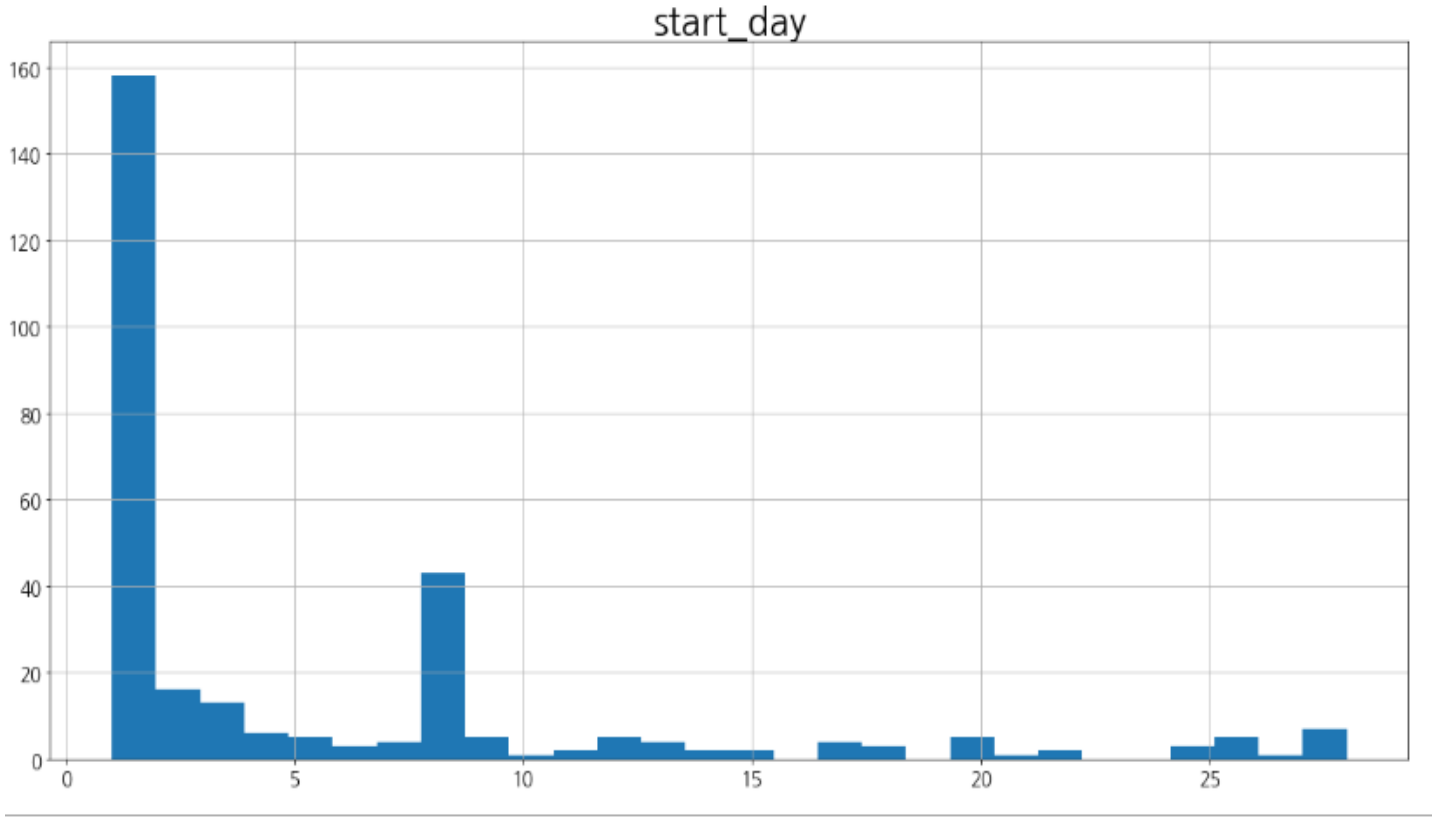
이탈 시기와 상관 없이 1일에 최초 등장한 유저의 비율이 압도적으로 많다.

1일 등장 유저가 전체 데이터셋의 대부분을 차지하기 때문에 나타나는 현상이다.

1일 등장 유저가 '이탈률' 자체는 낮지만, 반대로 '실제 이탈 수 ' 의 대부분을 차지하기 때문에 1일 등장 유저에 대한 이탈 예측이 중요하다고 생각했다.

Week1에 이탈한 유저는 최근에 유입된 신규 유저의 비율이 상대적으로 매우 높다.

Amount_spent 상위 집계

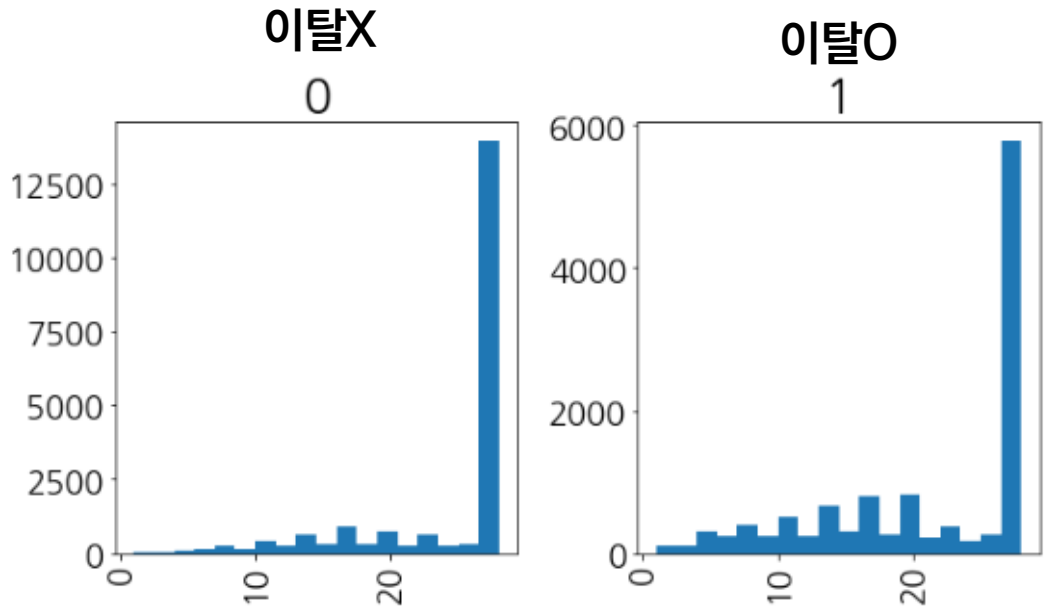


과금액 상위 3% 기준했을 때, 첫 로그인이 1일인 유저가 과반을 차지함.

과금액의 상당수를 차지하는 '고과금' 유저의 대부분은 첫로그인이 1일인 유저이다.

1일날 등장하며 amount_spent가 높은 유저를 잘 선별할 수 있다면 기대이익의 관점에서 긍정적일것이다.

이탈 여부에 따른 연속 로그인 수



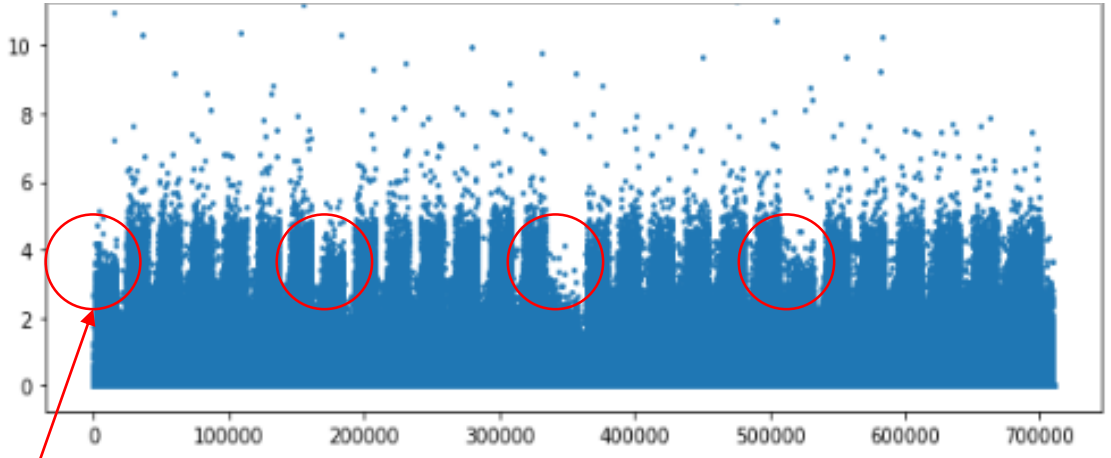
연속으로 로그인한 날의 수를 통해 유저가 얼마나 습관적으로 게임에 접속하는지 평가할 수 있다.

이탈을 한 경우에 연속 로그인 수가 적은 경우가 더 많다.

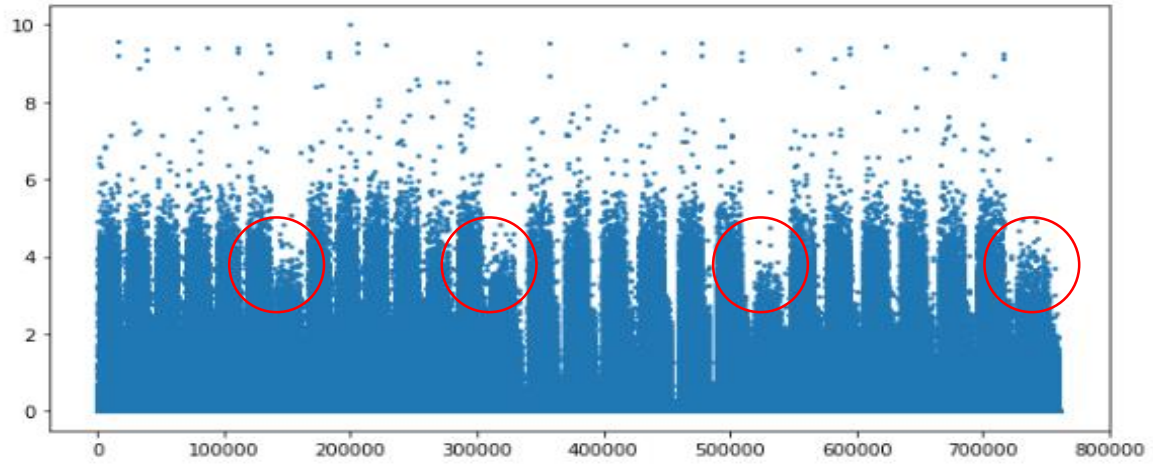
집계 시간 단위 결정(일 혹은 주)을 위한 EDA

Dataset마다 시작 요일이 다른 것으로 생각됨(서버 업데이트)

Train과 Test1의 요일별 산점



Test2의 요일별 산점



매주 수요일 서버 업데이트

"주별 집계를 사용하는 것이 특정 요일의 서버 업데이트로 인한 데이터 분포 변화에 영향을 받지 않음"

EDA와 분석 목적을 고려한 방향 설정

모델링 방향성

'과소평가' 보다는 '과대평가' 하게 모델을 만든다.

기대 이익(score) 을 최대화 하기 위해 이탈/과금의 threshold 하향 조정한다.

타겟인 '이탈일' 및 '평균과금액' 의 분포는 정규성을 띄지 않기에 비모수 모델로 예측한다.

1일 첫등장 유저에 대한 개별 모델

1인 등장 유저는 온전하게 모든 정보를 가지는 유저들이다. 따라서 모델링 시, 유저간의 특징을 온전히 잘 비교할 수 있다.
또한 총 유저의 대부분을 차지하며 과금액 상위 과반을 차지하는 유저들 인 만큼 다른 모델로 접근할 필요가 있다.

03 Feature Engineering 데이터 전처리

28일치

Acc_id	Char_id	Playtime_1	Playtime_2	...	...	...	...	Playtime_27	Playtime_28
A	1	0.1324	0.3461	...	...	...	...	0.4578	0.4262
	2	0.6325	0.9714	...	...	...	...	1.3423	0.1314
	3	0.1341	0.4234	...	...	...	...	1.0021	0.7533

유저별 집계

주별 집계

전체 집계

Acc_id	Playtime_week1	Playtime_week2	Playtime_week2	Playtime_week2	Total_playtime
A	0.1324	0.3461	...	...	...
	0.6325	0.9714	...	...	...
	0.1341	0.4234	...	...	...

03 Feature Engineering 주요 변수 생성

activity

- conseq_nolog : 연속 미 접속 횟수의 max
- Login_cnt : 총 로그인 횟수.
- Avail_day : 총 로그인 가능 횟수.(29- 첫 등장일)
- First_app : 첫 등장일.
- total_soloexp : 총 솔로 사냥 경험치 획득량
- Total_fishing : 총 낚시활동 시간
- Per_fishing : 전체 playtime 중 낚시 활동 시간 비율
- Maxplay_cnt : 28일중 24시간 전부를 게임한 날의 수.
- ….

payment

- total_pay_amount : 총 과금액 합.
- First_pay_from_start : 첫 등장일로 부터 첫번재 결제일 까지 걸린 시간
- Mean_pay_amount : 평균 과금액 (총 과금액/ 과금한 날 수)
- Pmday_playday_ratio : 실제 접속한 날 대비 과금한 날 비율
- Pay_yn : 과금 유무
- Days_since_last_pay : 마지막 과금일 로부터 경과일
- Pay_day_cnt : 과금한 날 횟수
- Max_pay_amount : 가장 과금액 이 높은 금액

etc

- combat_login : 전투 이력 있는 날 수
- change_level_avg : 레벨의 변화가 있는 날 수
- pl_play_char_cnt : 대표 캐릭터의 혈맹원들의 평균 접속수
- Pl_temp_cnt : 대표 캐릭터의 혈맹원들의 단발성 전투 평균
- Total_num_opponent : 총 전투 상대 캐릭터 수
- Most_trade_item : 가장 많이 거래된 아이템 (범주형)
- Most_trade_time : 가장 많이 거래된 시간
- Per_buy : 총 거래횟수 중 구입한 거래의 비율
- ……

4_week

- login_cnt_4 : 4주차 로그인 횟수
- Total_moneychange_abs_4 : 4주차 게임 머니 절대 변화량
- Trade_cnt_4 : 4주차 구입/판매 총 횟
- Questexp_4 : 4주차 퀘스트 획득 경험치 총합
- Total_privateshop_4 : 4주차 총 개인상점 시간 합
- Per_soloday_4 : 4주차 접속일 대비 개인 사냥 비율
- Indiv_trade_cnt_4 : 4주차 개인거래 횟수 총합
- Char_cnt_4 : 4주차 사용 캐릭터 총합
- ….

04 Modeling 데이터 전처리

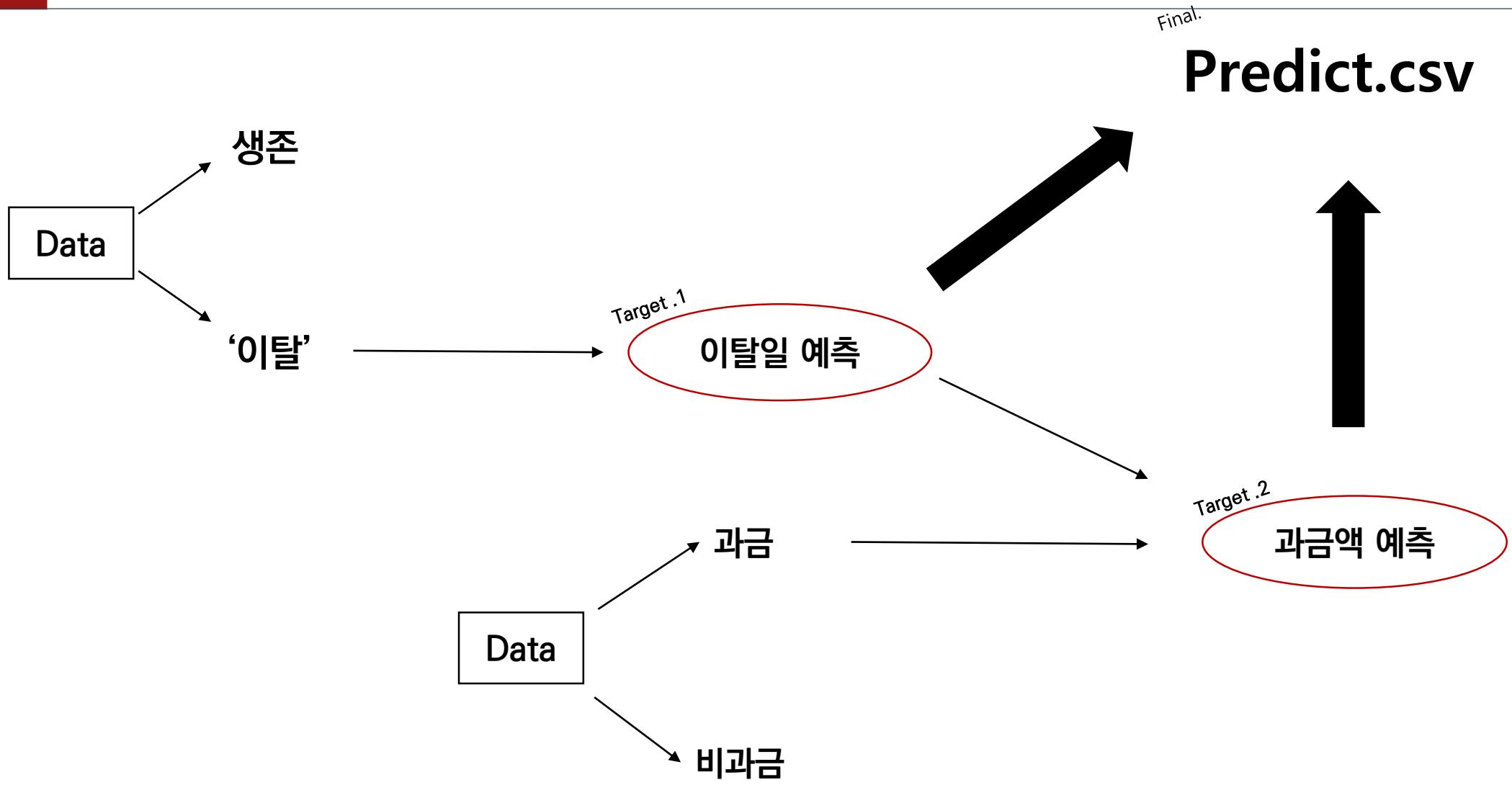
1일 등장 모델

- 1일에 등장한 사람들 데이터를 기준으로 만든 모델.
- 1일에 등장한 사람들 간 차이에 더 집중하기 위하여 만든다.
- 적용 대상 : test데이터 중 1일에 등장한 유저들.

일반 모델

- 전체 데이터를 기준으로 만든 모델
- 전반적인 데이터의 차이를 보기 위한 모델
- 적용 대상 : test데이터 중 1일에 등장한 유저 제외 전부.

04 Modeling 데이터 전처리

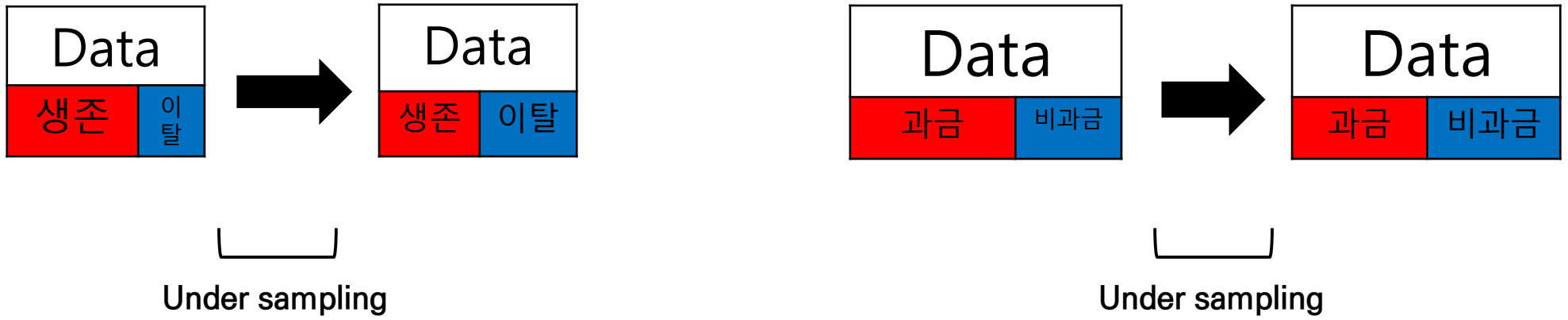


Under-sampling

- 데이터 클래스 비율이 너무 차이가 나면(highly-imbalanced data) 단순히 우세한 클래스를 택하는 모형의 정확도가 높아지므로 모형의 성능 판별이 어려워진다.

- 정확도(accuracy)가 높아도 데이터 갯수가 적은 클래스의 재현율(recall-rate)이 급격히 작아지는 현상이 발생할 수 있다.

-> Under sampling 으로 비율을 동등하게 구성.

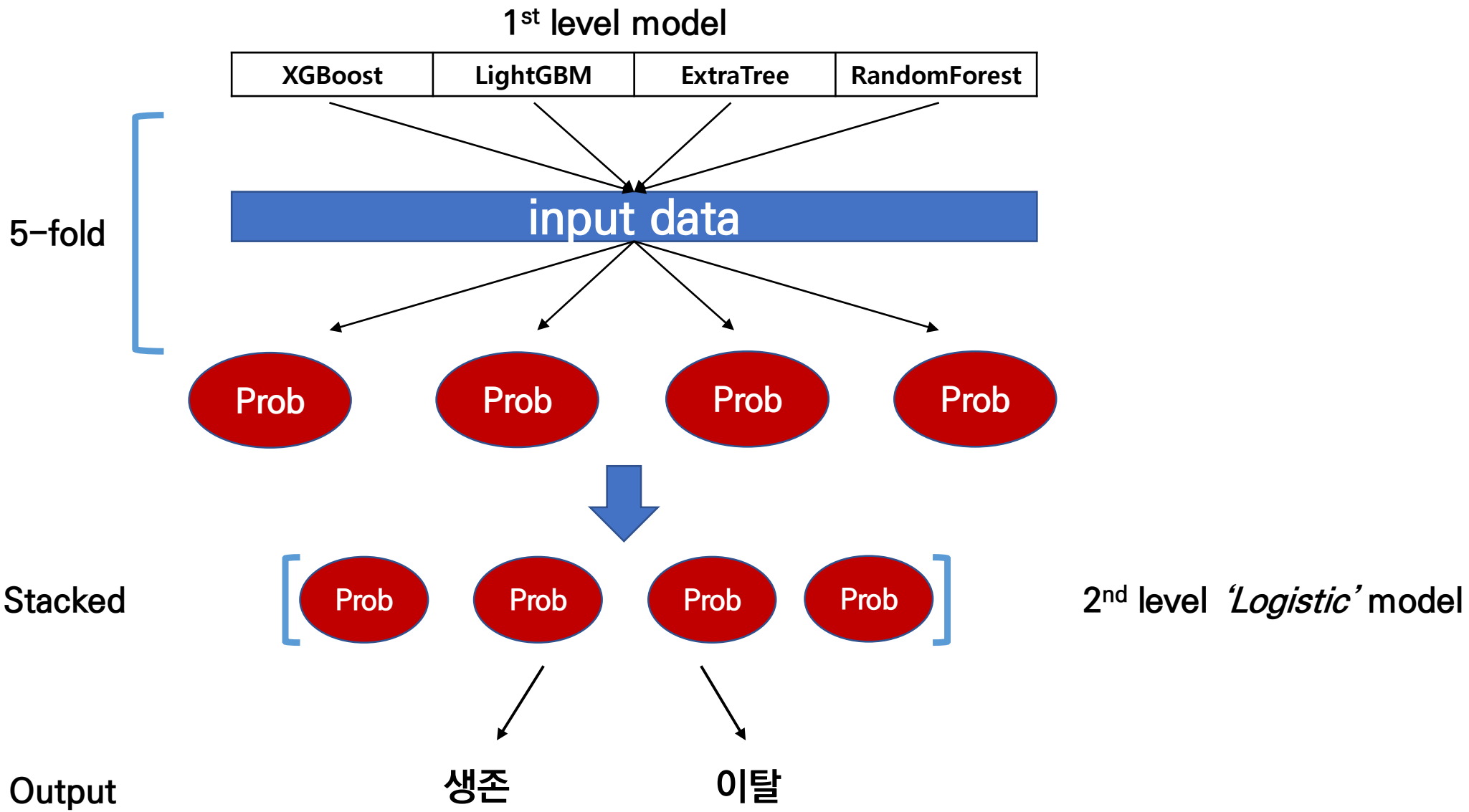


04 Modeling 데이터 전처리

Base-line Model.

XGBoost	- 유연성이 좋으며, 스스로 가지치기를 통해 과적합이 비교적 덜 발생 - 다른 모델과 덧붙이기 용의해 ensemble기법에 자주 사용됨
Light GBM	- 트리 기반의 모델로, 빠른 속도와 높은 효율성이 장점. - 대용량 자료 처리에 용의
Extra Tree	- 트리 기반의 모델로, 무작위(random)성이 높음 - 트리를 비 복원 추출하여 구성.
Random Forest	- 높은 정확성과 간단하고 빠른 알고리즘 - 무작위를 통한 높은 일반화 성능

04 Modeling 데이터 전처리

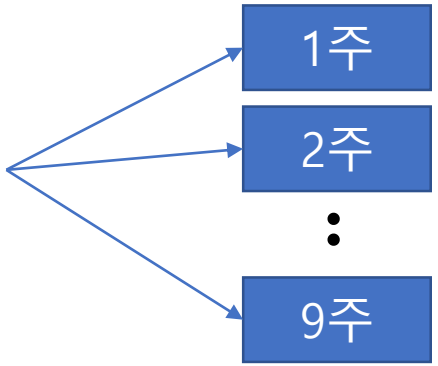


04 Modeling 데이터 전처리

이탈일 예측.

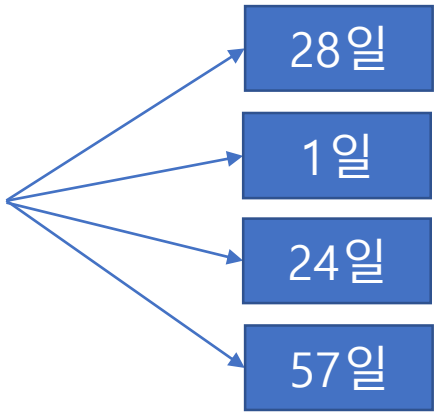
이탈 주차

Model.1
XGB : multi-class classification.



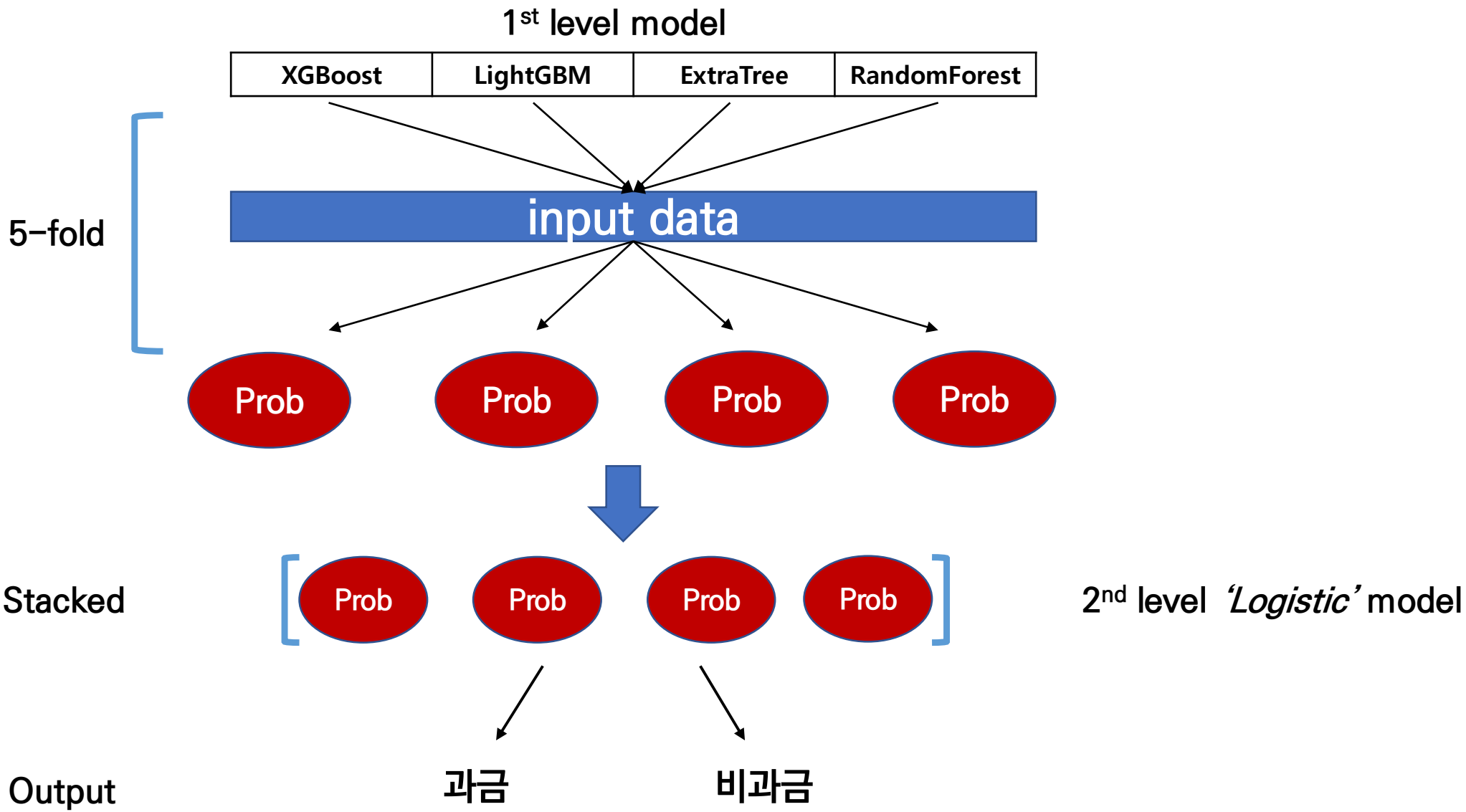
이탈일

Model.2
Neural-Network



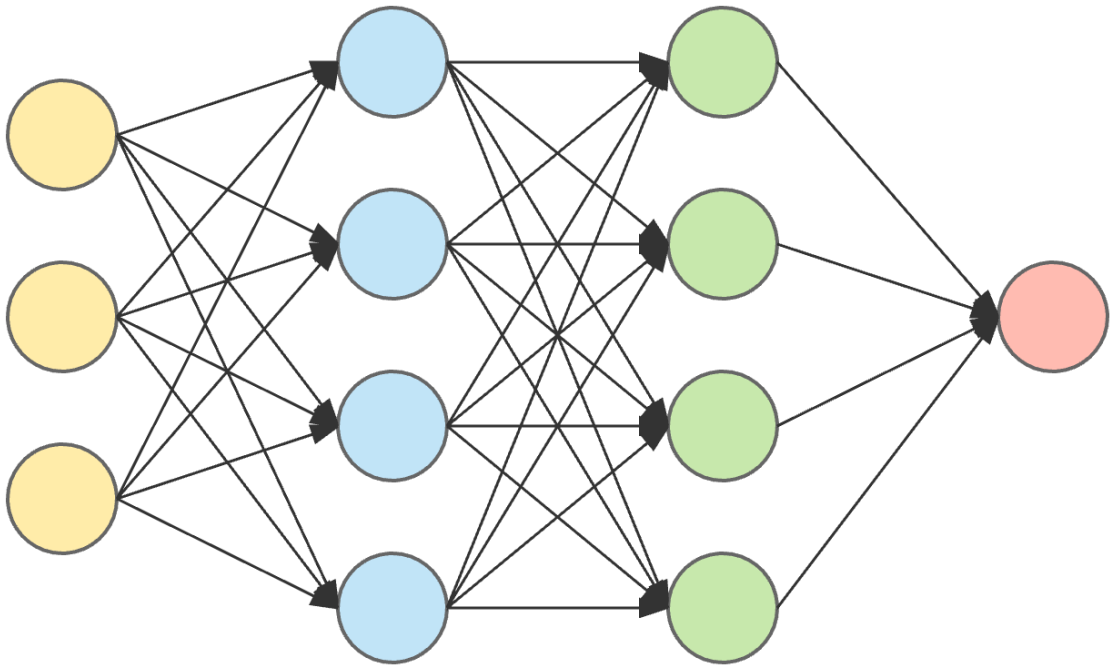
Acc_id	Survival_time
2435	39
12	2

04 Modeling 데이터 전처리



04 Modeling 데이터 전처리

Neural-Network



- 과금액의 분포는 과금이 증가할수록 수가 감소하는 포아송(또는 음이항) 분포의 성질을 가진다.
- 일반적인 선형 회귀의 방법으로는 해당 분포의 회귀에 대한 정확한 예측이 어렵다
- 손실은 금액을 과대평가 했을 때 보다 과소평가했을 때 아주 크다.

N-N model
(with Custom-loss Function)

- (Max Score – predicted Score) 를 최소화 하는 함수.
- 과소 평가 시 손해가 더 큰 비대칭적인 loss 함수

감사합니다