2018 빅콘테스트 Analysis분야 챔피언 리그

‘블레이드앤소울’ 게임 유저 이탈 예측 문제

September 14, 2018

Team **추적자**: 오혜신, 배소현, 윤현근, 안태언, 손원철

Table of Contents

1. Introduction

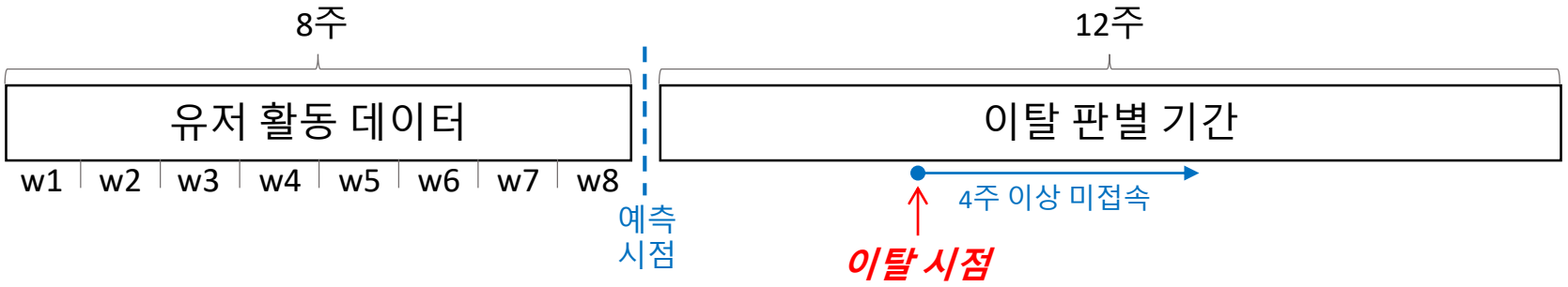
1.1 문제 및 평가기준 소개

- **게임 '블레이드 앤 소울' 유저 이탈 예측 모형**
  - 엔씨소프트의 MMORPG 게임 '블레이드앤소울' 유저의 활동데이터를 활용하여
    어떤 유저가 향후 게임서비스에서 이탈하는지, 언제 이탈하는지 예측하는 모델 개발

- 의의
  - 고객 이탈은 다양한 도메인의 CRM에서 중요하게 다루는 문제
  - 게임 데이터: 현실 사회와 유사한 가상 세계에서의 인간 행동에 대한 고품질 데이터가 제공되는 context

1.1 문제 및 평가기준 소개

- 문제: 게임 유저의 향후 이탈 여부 및 시점 예측
- '이탈'의 정의: "**4주 이상 접속하지 않으면 이탈로 판단한다.**"
  - 제공 데이터 시점 이후 12주 동안의 접속 이력으로 판단



- Target label: 이탈 여부, 이탈 시점에 따른 4개 클래스

1.1 문제 및 평가기준 소개

- 평가: 예측 성능 + 서류 심사
  - 예측 성능: F1 score
    - 각 클래스별 precision과 recall을 계산한 후 전체에 대한 조화 평균
    - $$F1 = \frac{8}{(\frac{1}{Week_{PR}}+\frac{1}{Week_{RC}}+\frac{1}{Month_{PR}}+\frac{1}{Month_{RC}}+\frac{1}{2Month_{PR}}+\frac{1}{2Month_{RC}}+\frac{1}{Retained_{PR}}+\frac{1}{Retained_{RC}})}$$
  - 서류 심사
    - 데이터 전처리 및 학습 알고리즘에 대한 설명
    - 모델 해석 및 이탈 원인 분석 (논리적인 접근, 적절한 시각화)

1.2 데이터 소개

- 데이터 보안상의 이유로 모든 id는 암호화처리, 모든 통계량은 표준화처리되어 있음
- 데이터 규모
  - Train data: (계정 아이디 기준) 10만 명의 게임 활동 데이터
  - Test data: (계정 아이디 기준) 4만 명의 게임 활동 데이터
- 데이터 구성
  - 주요 활동 정보 "activity": 게임 내 주요 활동량을 유저별 1주일 단위로 집계
  - 결제 정보 "payment": 사용자의 결제 정보를 1주일 단위로 집계
  - 사회관계 정보 "party", "guild", "trade": 유저간 상호작용 및 사회관계에 대한 정보 (예측 대상이 아닌 유저 포함)
- 제공 파일

Train data set

- Train_label
- Train_activity
- Train_payment
- Train_party
- Train_guild
- Train_trade

Test data set

- Test_activity
- Test_payment
- Test_party
- Test_guild
- Test_trade

2. EDA & Feature Engineering

2.1 EDA & Feature Engineering Overview

- Overview
  - 각각 다른 schema를 가진 data에서 예측 대상인 유저id(acc_id) 기준으로 하여 feature variable 생성

1) Activity, payment data의 경우

  - 한 유저가 week 별로 여러 개의 관측치를 가지고 있음

  ⇨ 이를 column마다 week별 변수로 확장(w1~w8, groupby하거 이들의 비율을 구해 변수 생성

2) Party, guild와 trade data의 경우

  - party와 guild의 경우 개인 유저 레벨이 아닌 그룹(party, guild) 레벨의 data
  - 전체 사회관계를 담기 위해 train id에 대해 sampling되어 있지 않음

  ⇨ party 멤버 id와 guild 멤버id에서 개별id를 추출하여 참여 횟수 등의 변수 생성

  ⇨ trade의 경우 전체 trade 리스트 중 train id가 구매/판매한 데이터만 이용해 변수 생성

  ⇨ trade와 party 전체 데이터에서 network를 구해 중심성 변수 생성

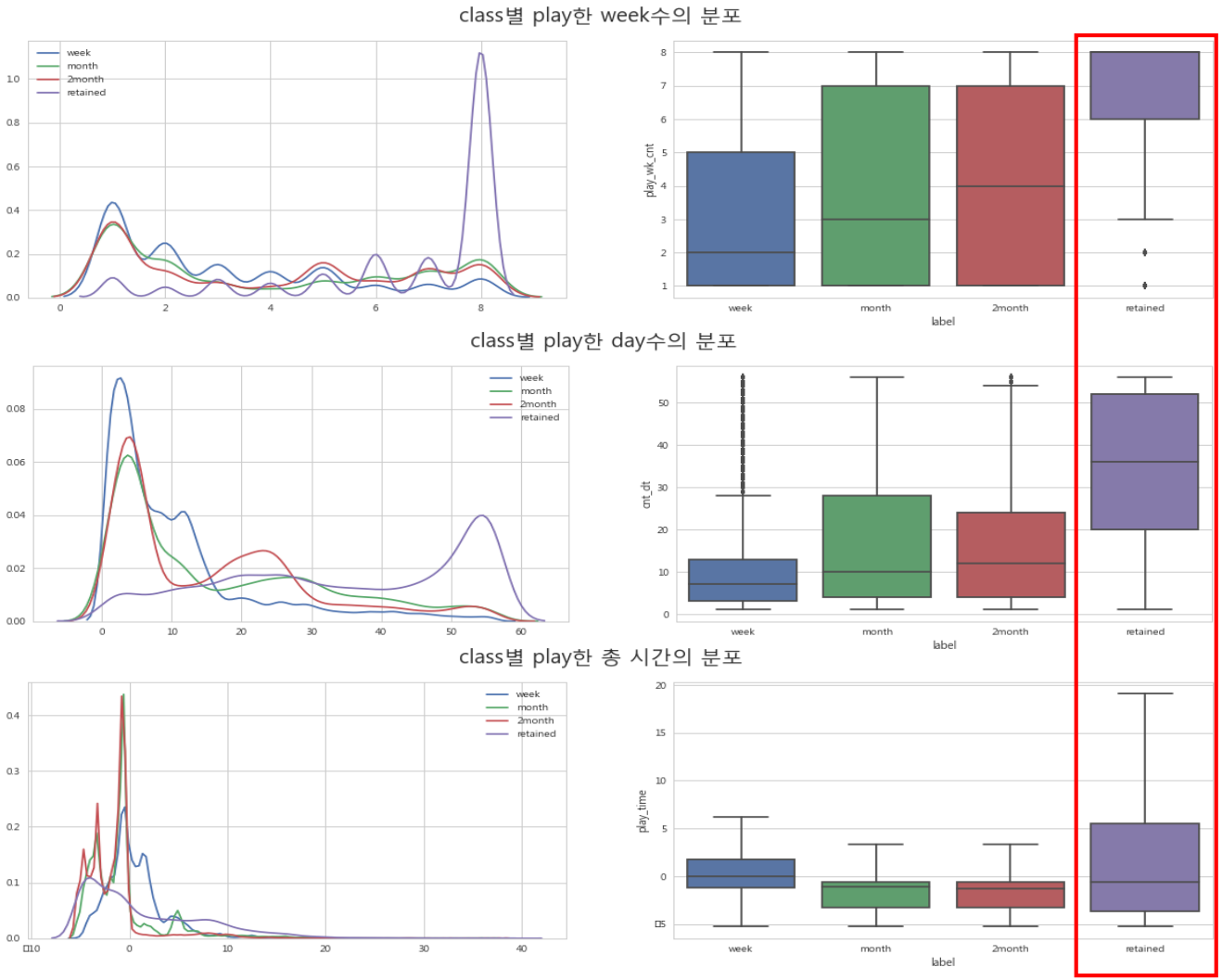
  - 현재 feature variables 총 536개
    - Modeling에서는 feature간 상관관계 등을 고려하여 선택 사용

Key Takeaways from EDA

1. *Retained 유저는 두드러지는 특징을 보인다*
2. *Week 유저 또한 두드러지는 특징을 보인다*
3. *Month와 2Month 유저는 구분하기 쉽지 않다*

2.3.1 EDA – Retained의 두드러지는 특징

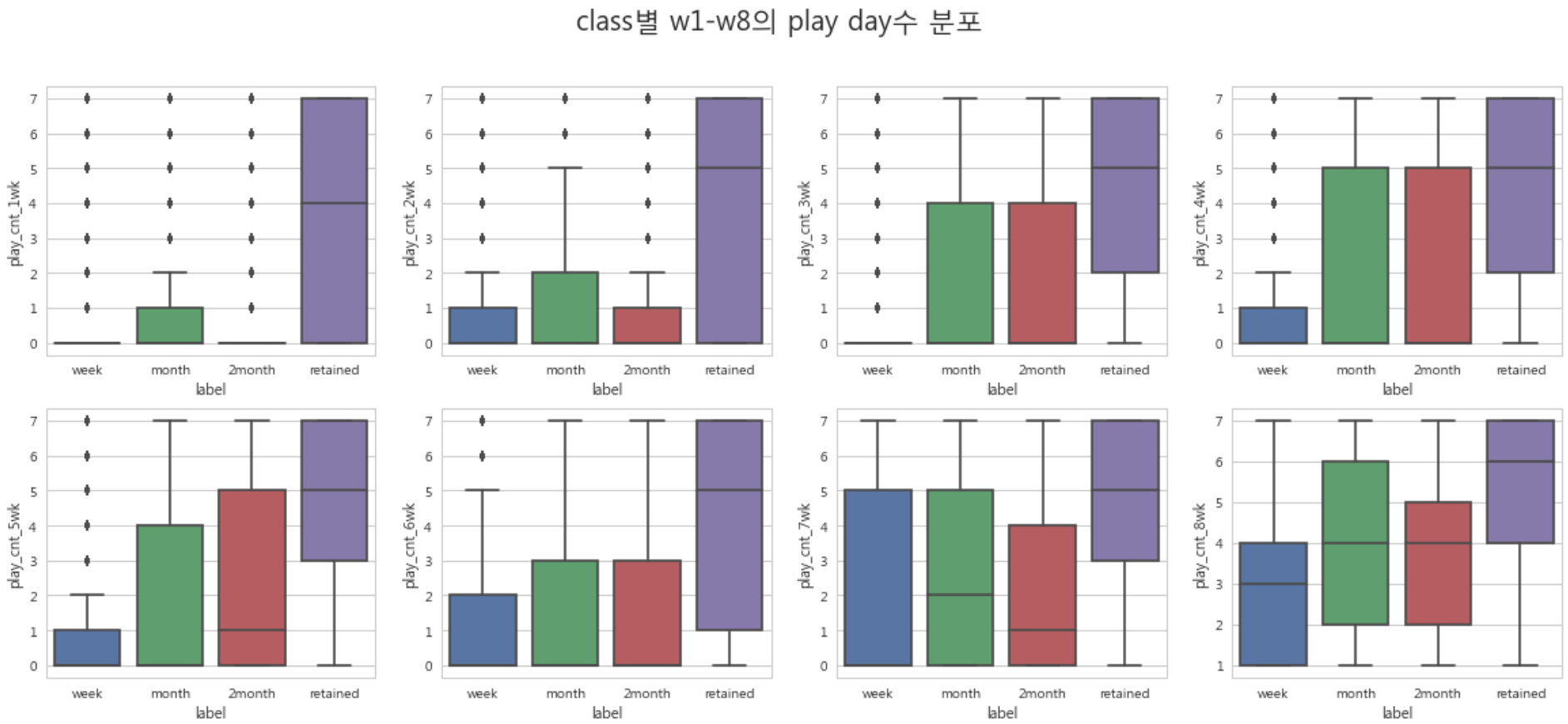
“play 시간과 횟수가 많은 유저는 잔존한다”



retained class

2.3.1 EDA – Retained의 두드러지는 특징

“8주 동안 꾸준히 자주 play한 유저는 잔존한다”



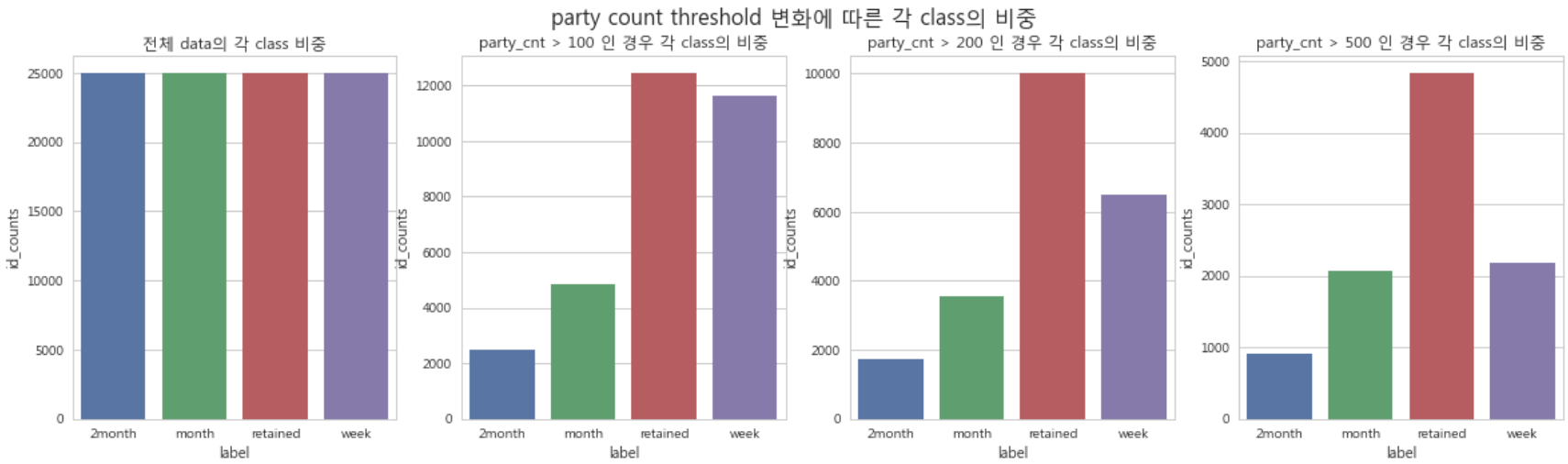
2.3.1 EDA – Retained의 두드러지는 특징

"결제하는 유저는 잔존한다"

retained class

2.3.1 EDA – Retained의 두드러지는 특징

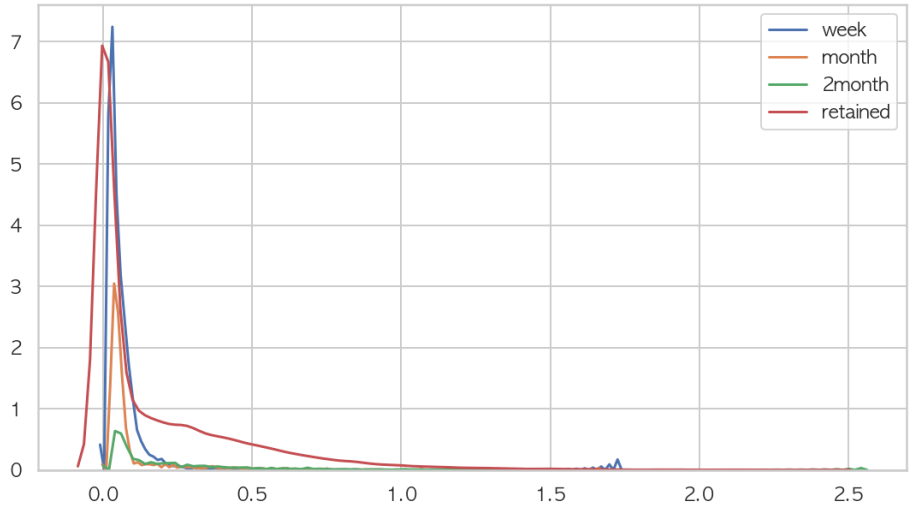
“Party에 많이 참여할수록 잔존 가능성이 높다”

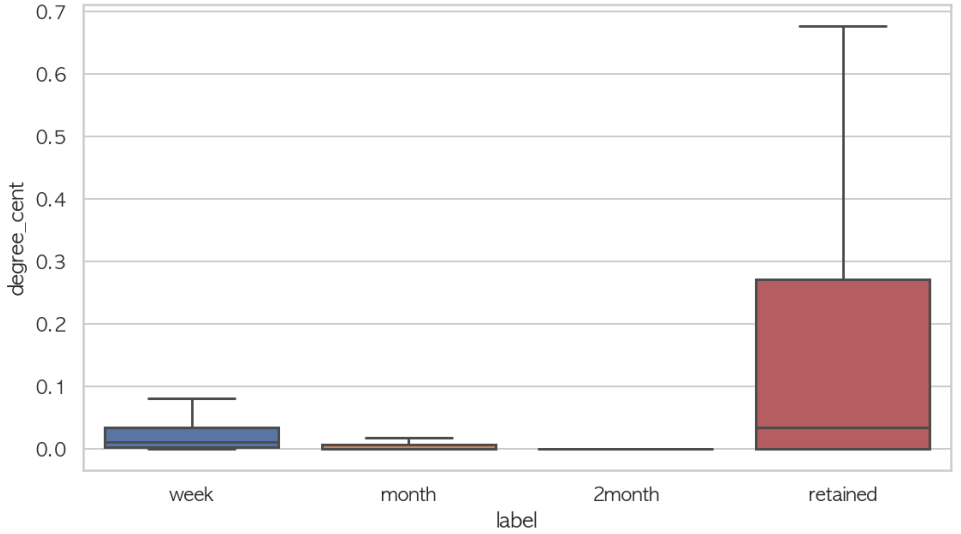


2.3.1 EDA – Retained의 두드러지는 특징

“Party network에서 중심성이 높으면 잔존 가능성이 높다”

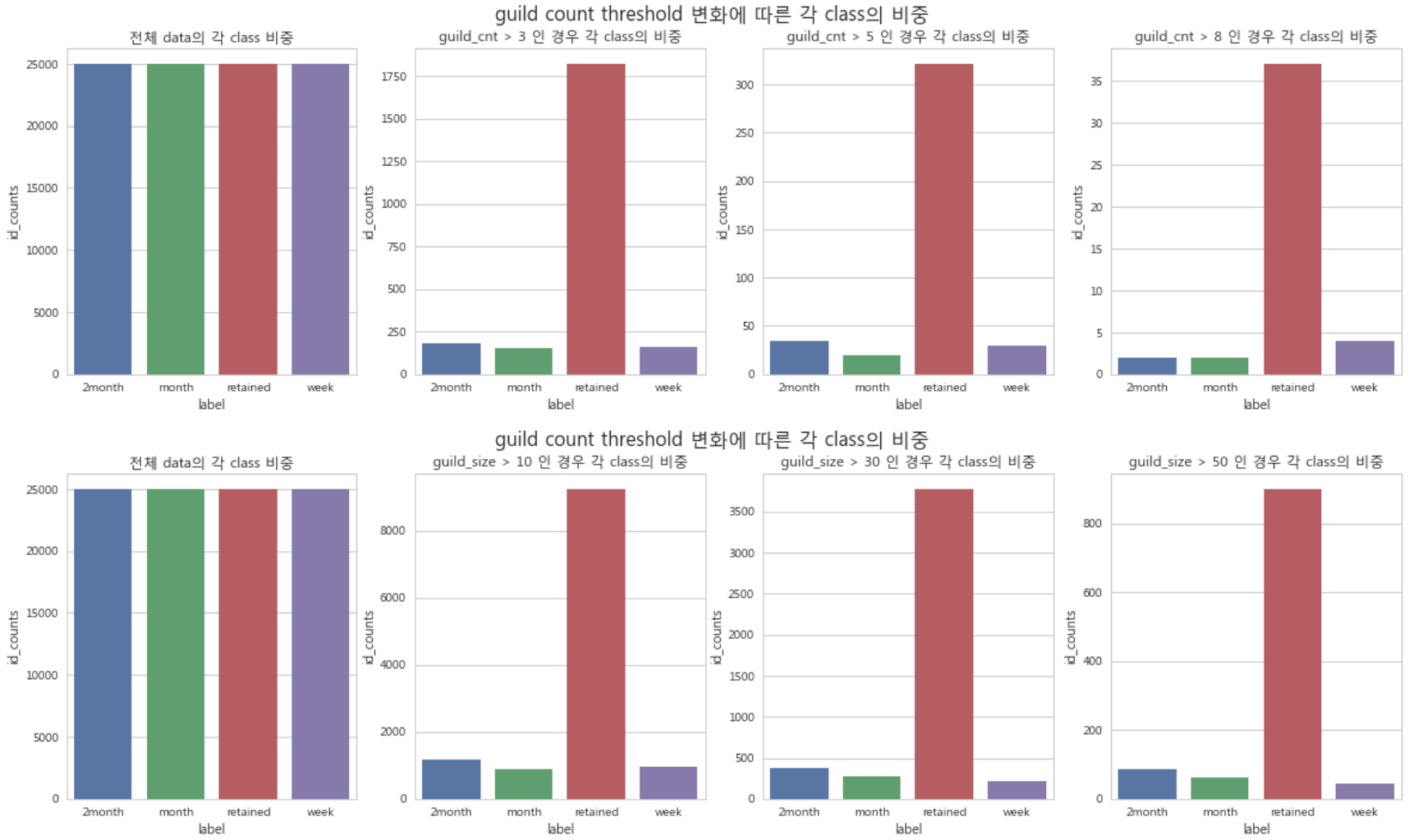
class별 party centrality의 분포





2.3.1 EDA – Retained의 두드러지는 특징

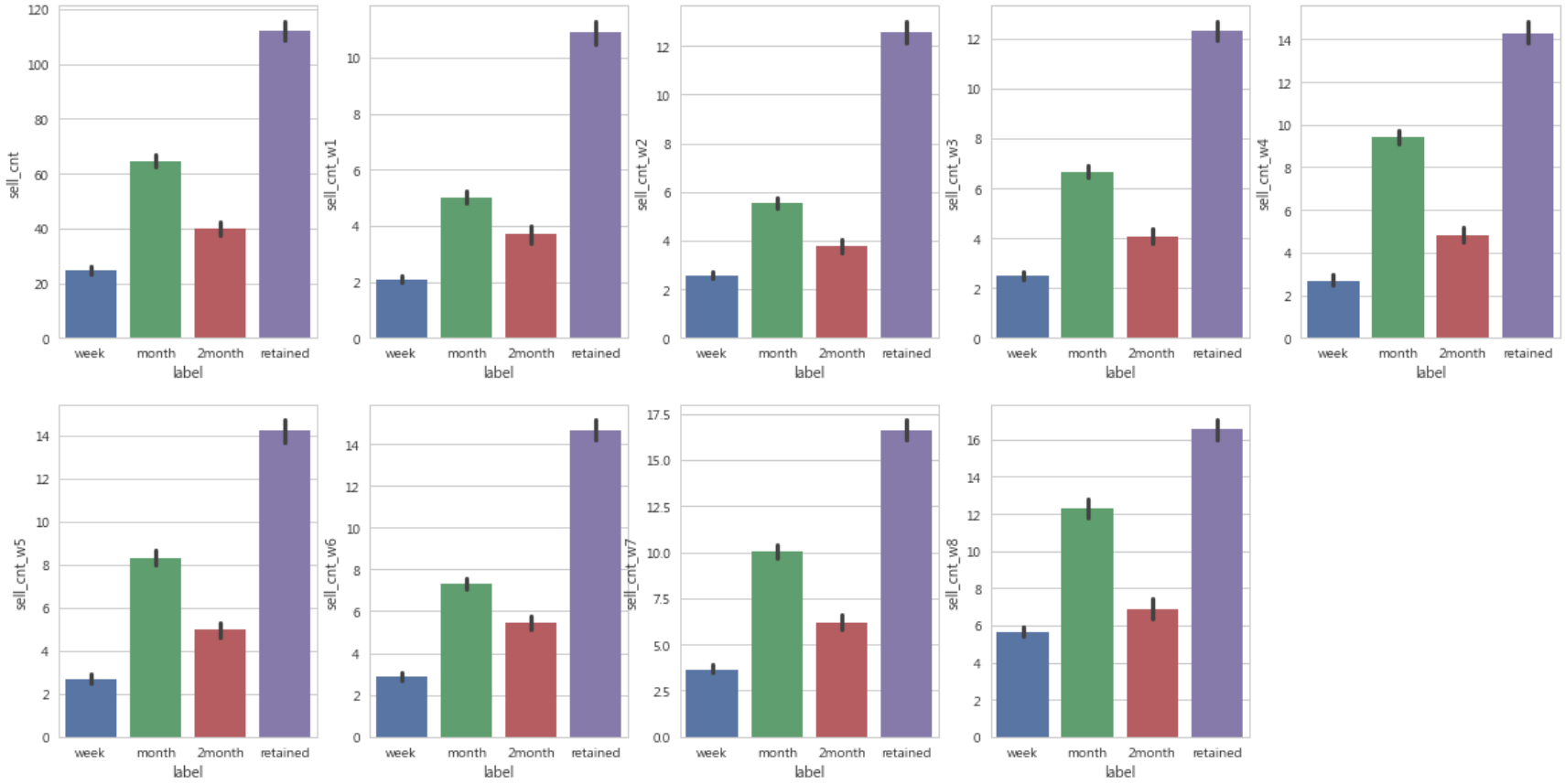
“Guild에 많이 가입할수록, 가입한 guild의 크기(멤버 수)가 클수록 잔존 가능성이 높다”



2.3.1 EDA – Retained의 두드러지는 특징

“유저간 주는 거래(판매)를 많이 하는 유저는 잔존한다”

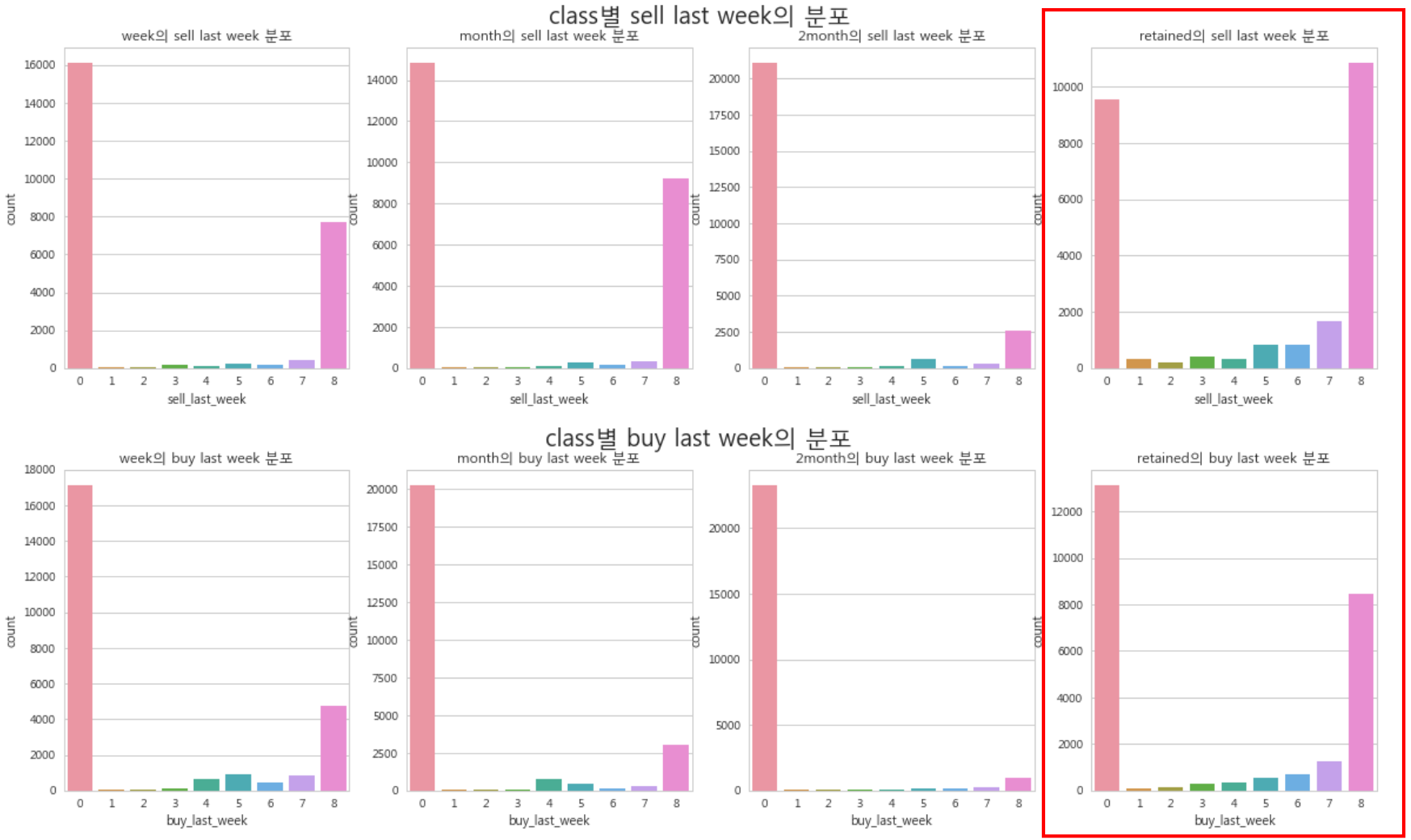
class별 주는 거래 횟수 (총횟수 & 주별 거래횟수)



2.3.1 EDA – Retained의 두드러지는 특징

"7-8주차까지 유저간 거래를 했던 유저는 잔존 확률이 높다"

retained class

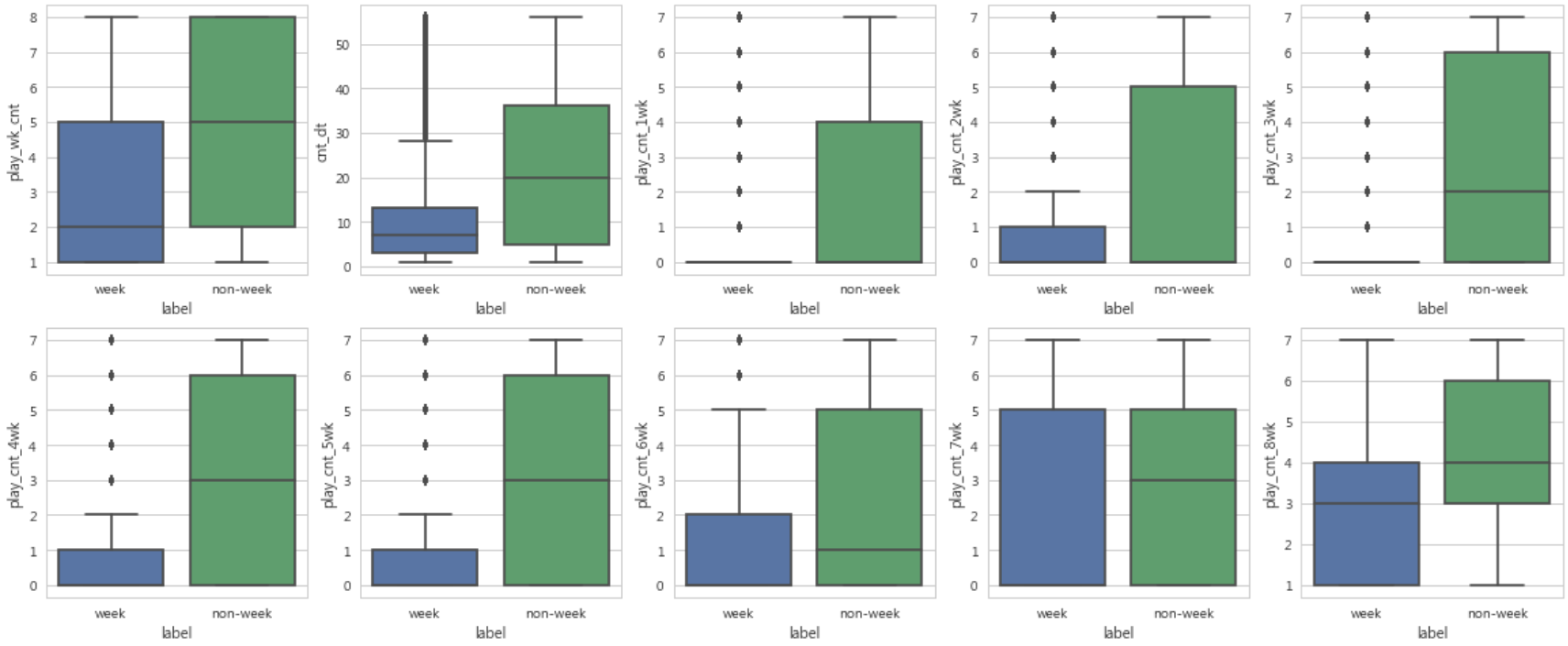


2.3.2 EDA – Week의 두드러지는 특징

"탈진하는 유저는 빨리 이탈한다"

week 유저는 play 횟수에 비해 play time이나 퀘스트 활동량이 많다

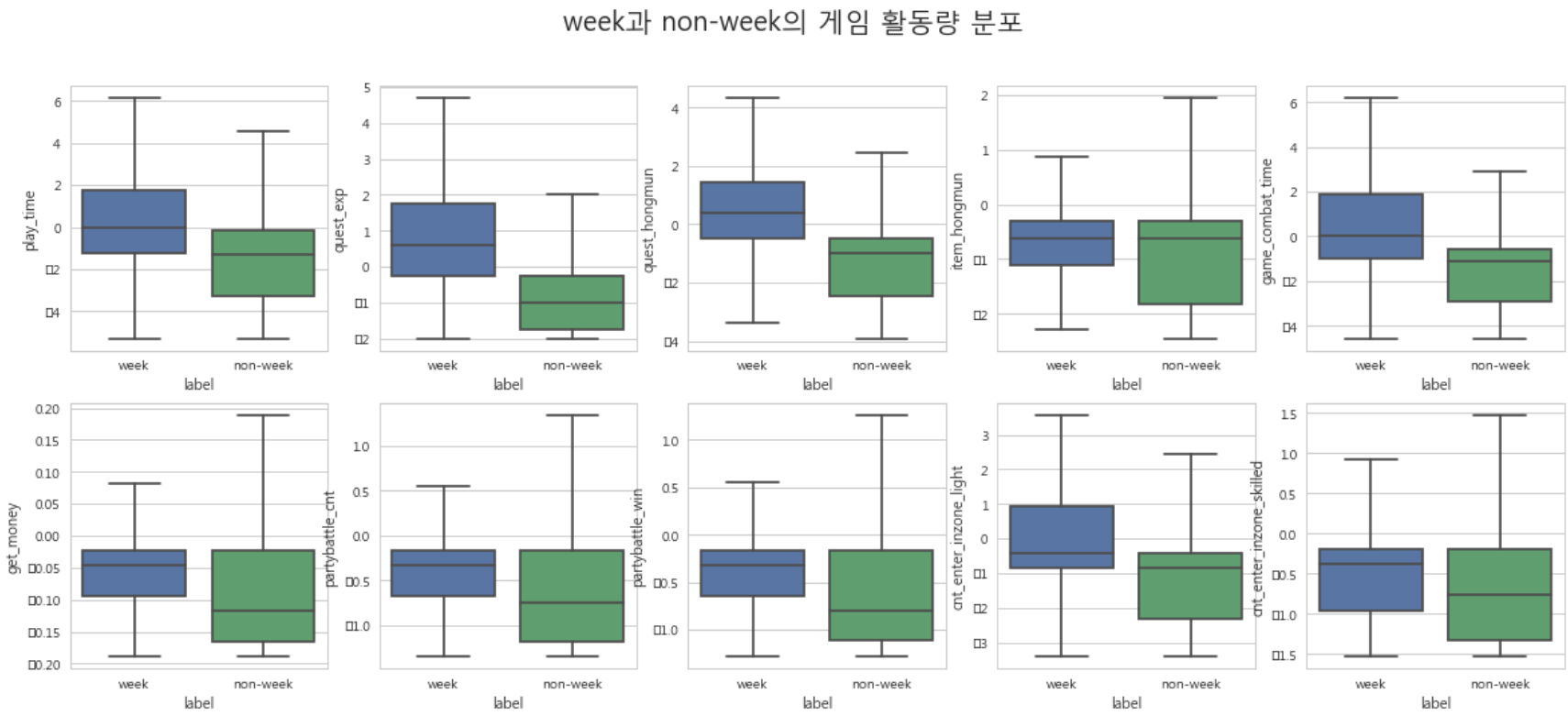
week과 non-week의 play횟수 분포



2.3.2 EDA – Week의 두드러지는 특징

"탈진하는 유저는 빨리 이탈한다"

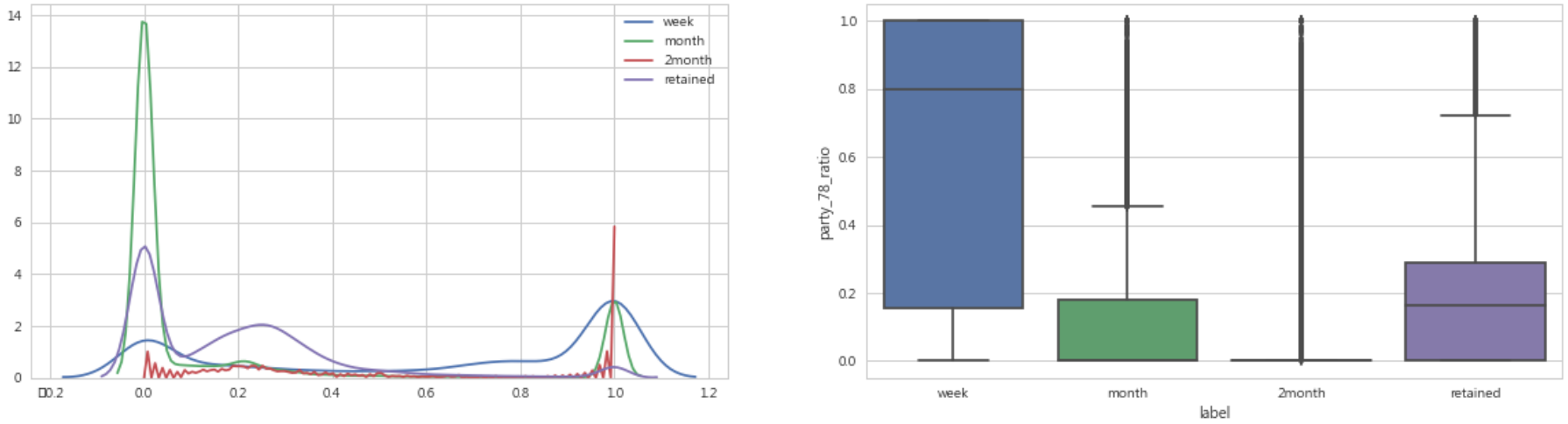
week 유저는 play 횟수에 비해 play time이나 퀘스트 활동량이 많다



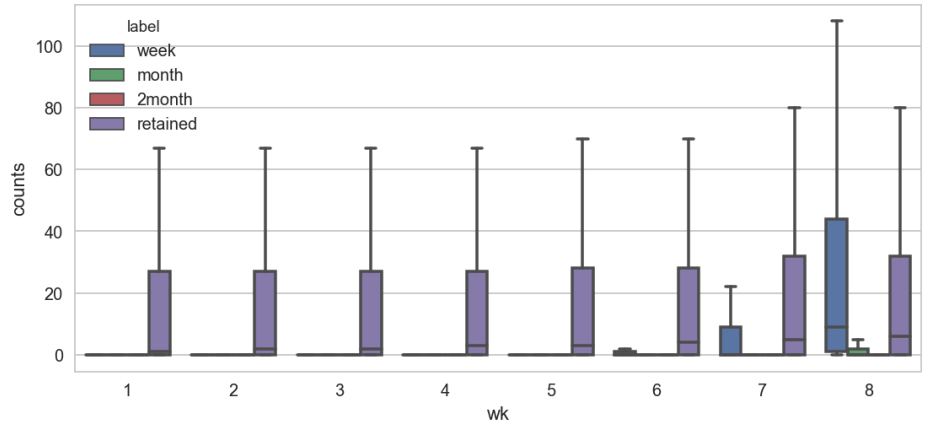
2.3.2 EDA – Week의 두드러지는 특징

"7-8주에만 몰아서 파티에 참여한 유저는 빨리 이탈한다"

class별 party_78_ratio의 분포

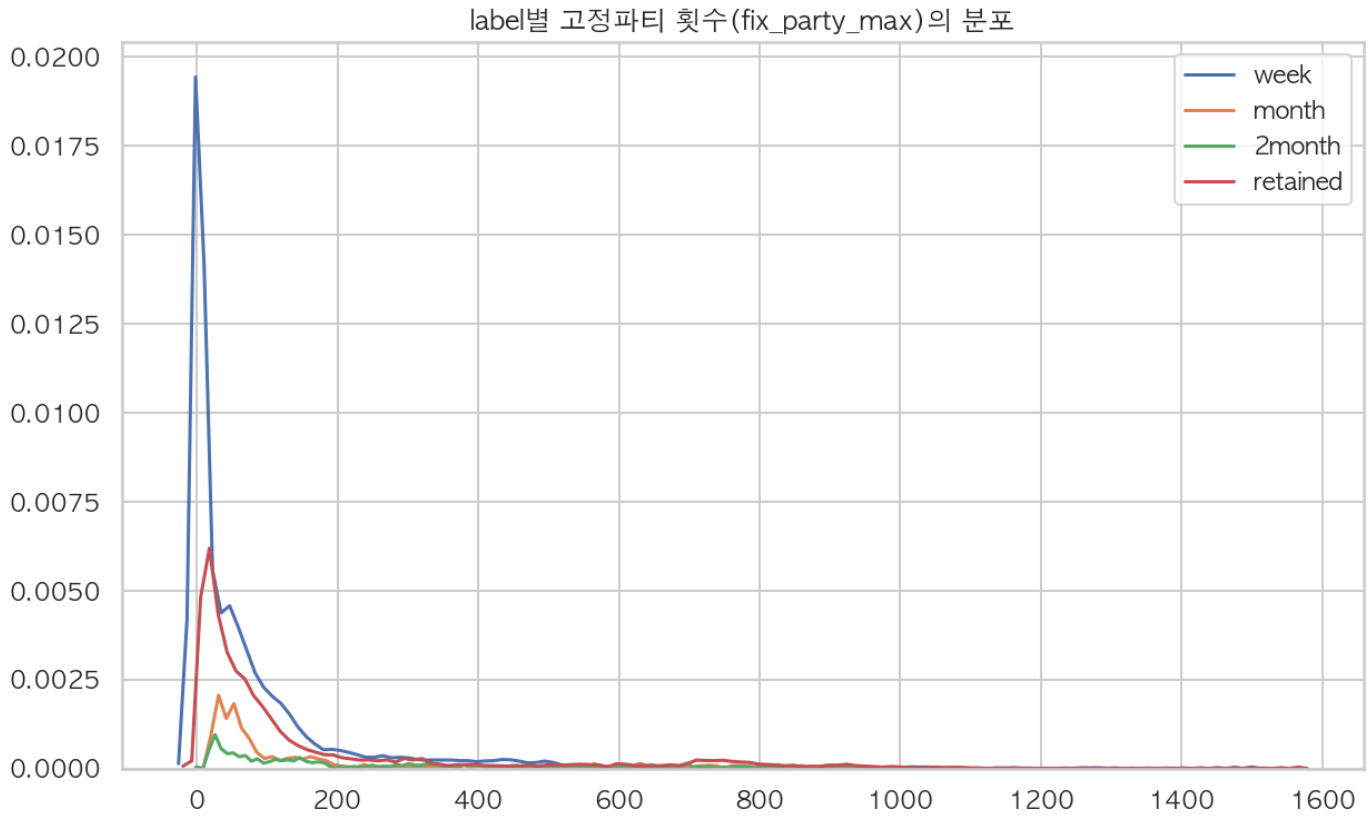


Class별 파티 참여 횟수의 주별 분포



2.3.2 EDA – Week의 두드러지는 특징

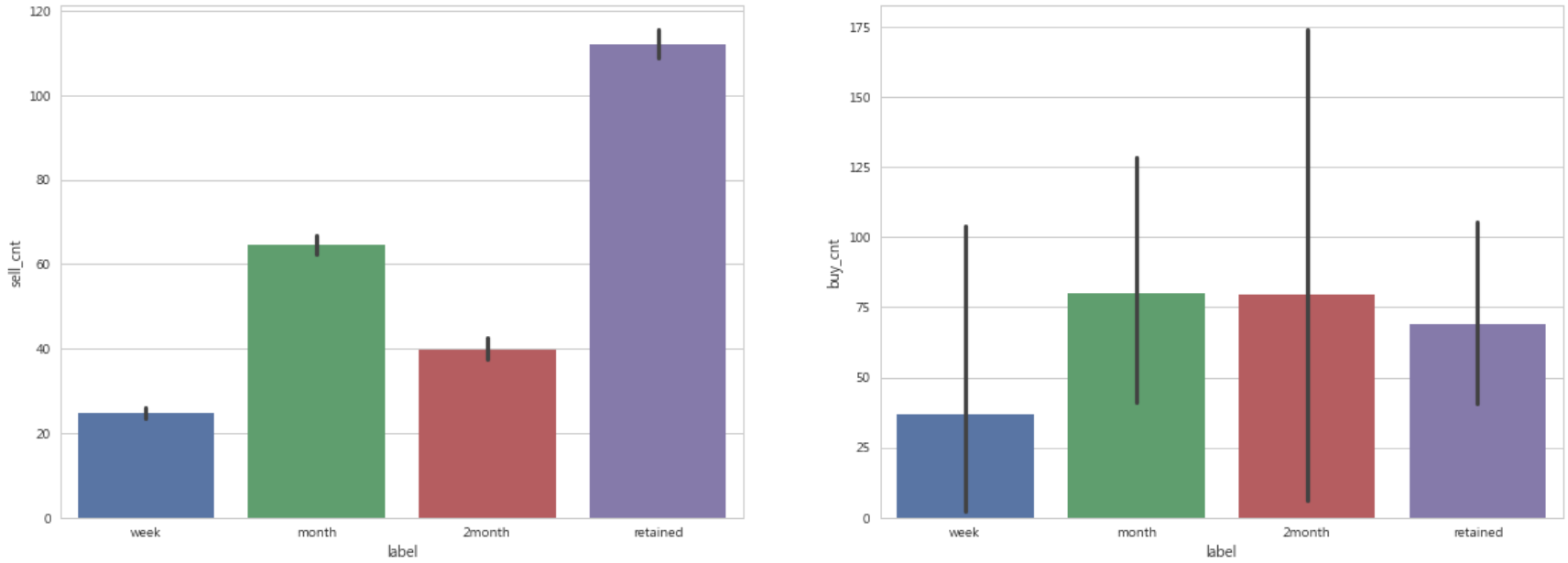
" 고정파티를 하지 않으면 빨리 이탈한다"



2.3.2 EDA – Week의 두드러지는 특징

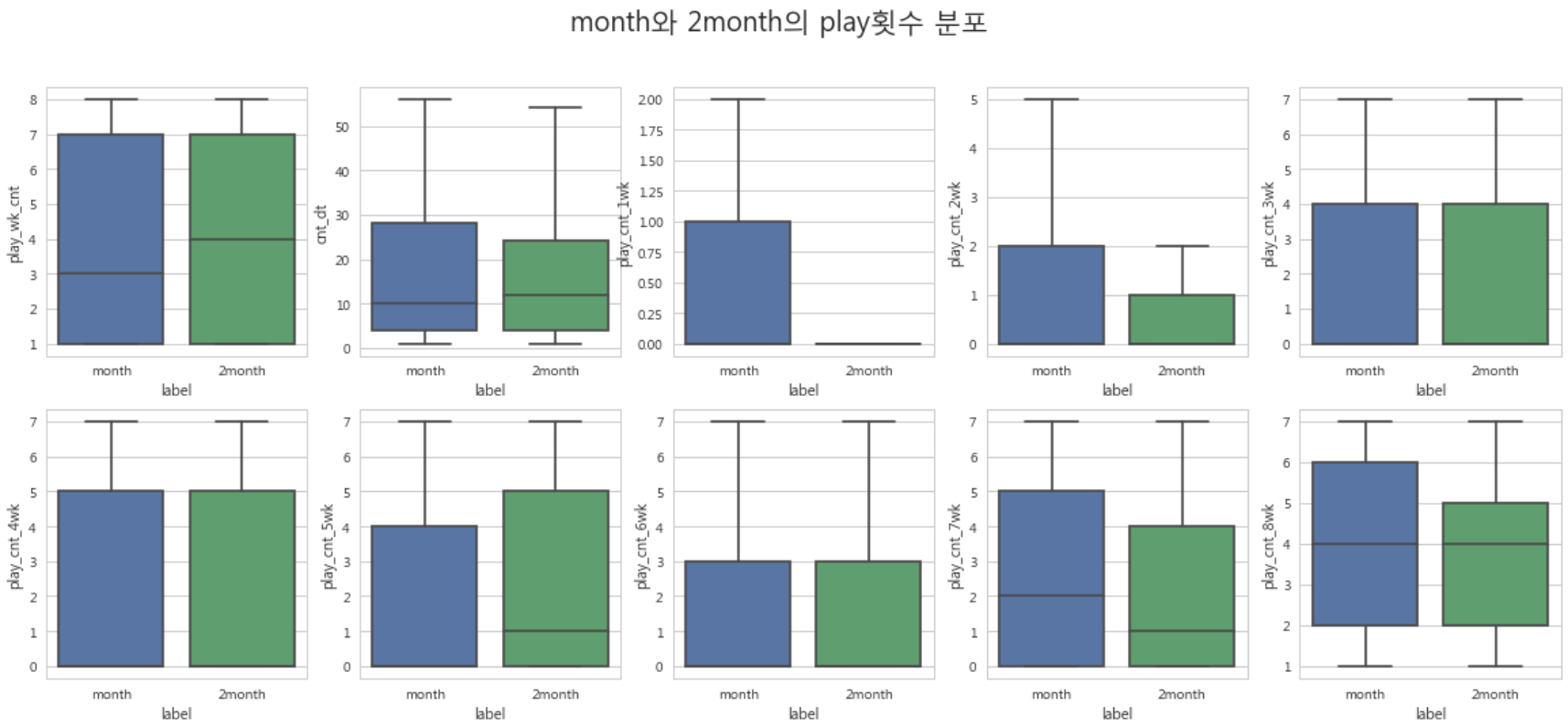
"유저간 거래에 참여하지 않는 유저는 빨리 이탈한다"

class별 유저간 주는 거래(sell), 받는 거래(buy) 횟수의 분포



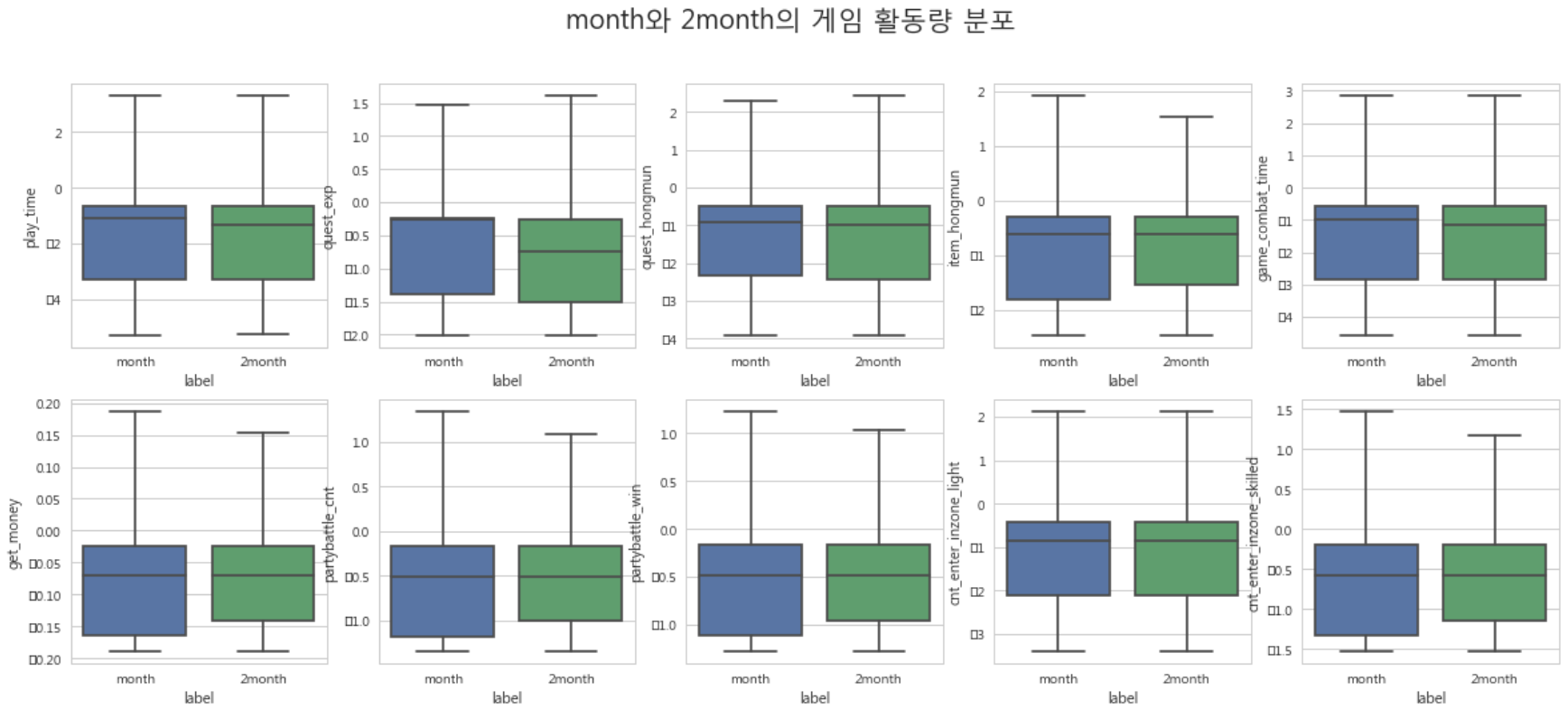
2.3.3 EDA – 구분하기 어려운 Month와 2Month

“month와 2month는 게임 이용 패턴에서 거의 차이가 나타나지 않는다”



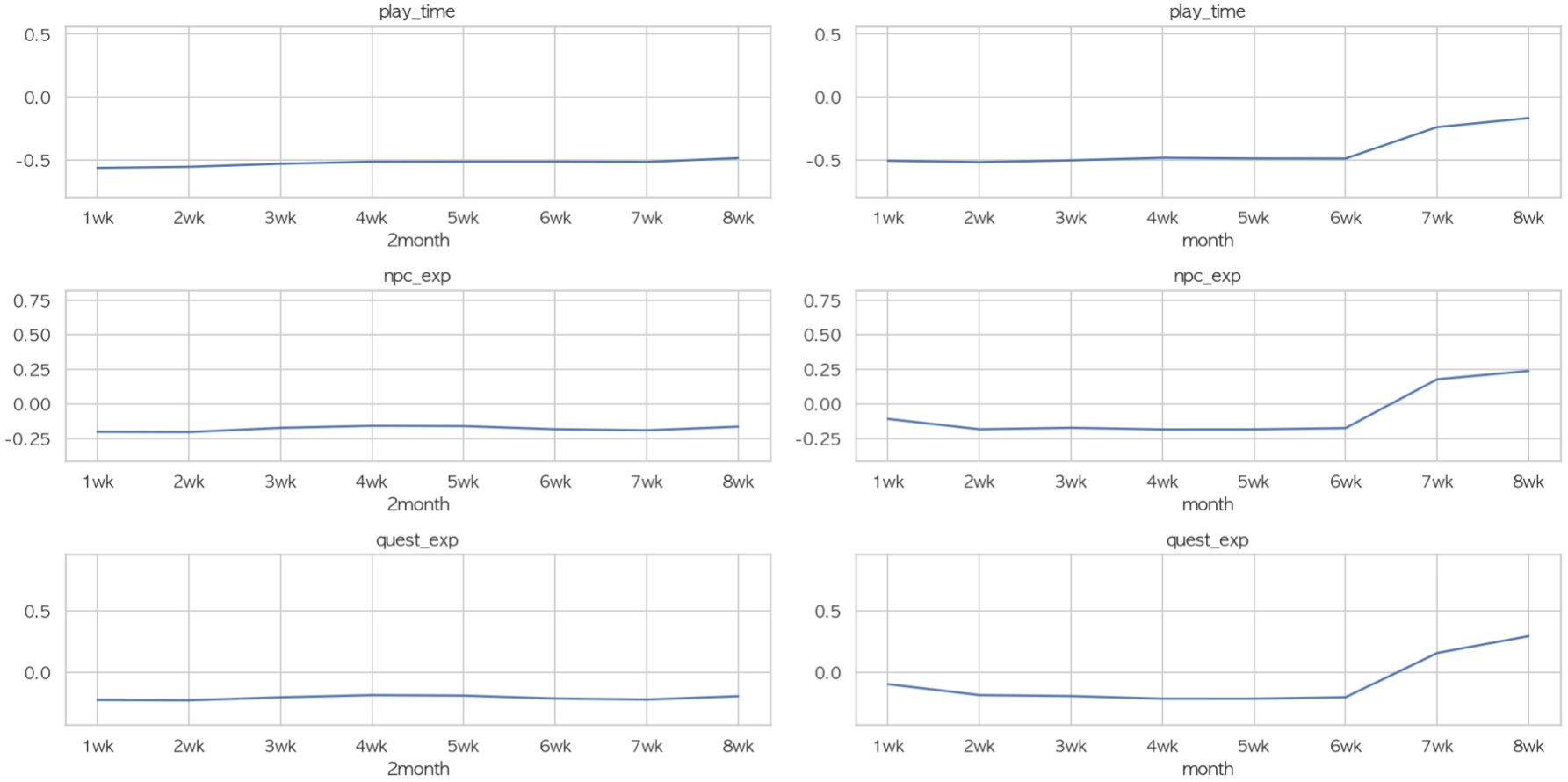
2.3.3 EDA – 구분하기 어려운 Month와 2Month

“month와 2month는 게임 이용 패턴에서 거의 차이가 나타나지 않는다”



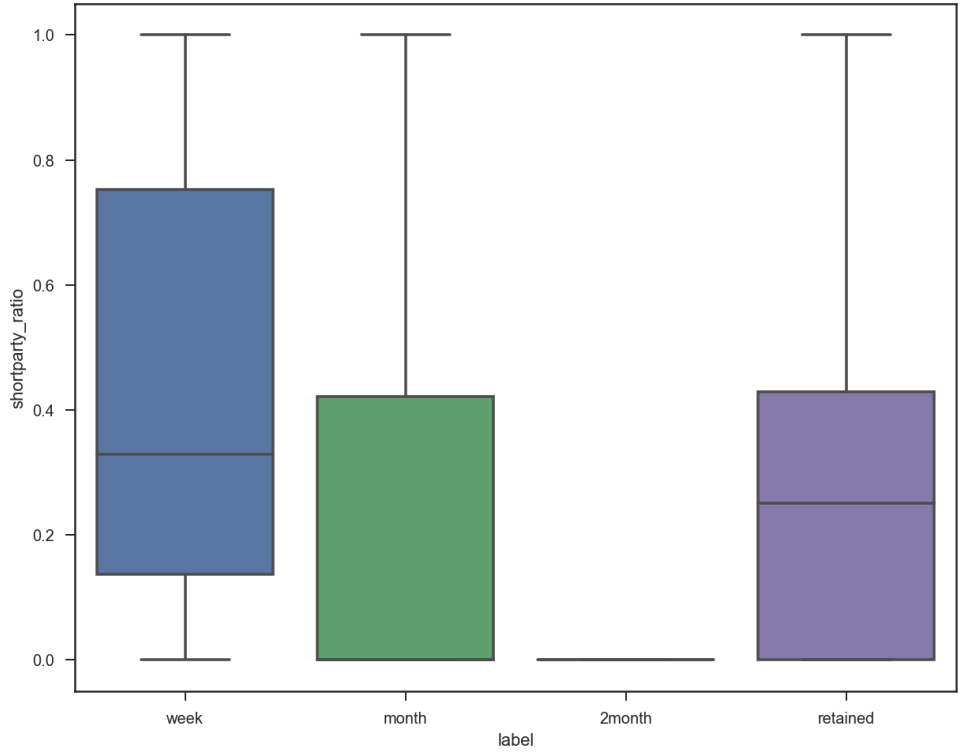
2.3.3 EDA – 구분하기 어려운 Month와 2Month

"주별 게임 활동량으로 나누어 살펴보면 month와 2month 사이의 약간의 차이가 나타난다"



2.3.3 EDA – 구분하기 어려운 Month와 2Month

“10분 이내에 종료된 파티에 참여한 횟수에서 2Month와 Month의 차이가 드러난다”



2.2 주제별 Feature Engineering - Activities

- Activity data: 유저의 인게임 활동 정보를 일주일 단위로 집계 (shape: 440,323 * 38)
- Data schema – *통계치는 모두 표준화되어 있음*

활동 주에 따라
한 id의 관측치가
여러 번 나타남

wk	활동 주(1-8)
acc_id	계정 아이디
cnt_dt	해당 주 접속 일 수
play_time	플레이시간 (초)
npc_exp	NPC 사냥 일반 경험치
npc_hongmun	NPC 사냥 홍문 경험치
quest_exp	퀘스트 일반 경험치
quest_hongmun	퀘스트 홍문 경험치
item_hongmun	아이템 홍문경험치
game_combat_time	전투 시간 (초)
get_money	재화 획득량
duel_cnt	결투 참여 횟수
duel_win	결투 승리 횟수
partybattle_cnt	전장 참여 횟수
partybattle_win	전장 승리 횟수
cnt_enter_inzone_solo	솔로 인던 입장 횟수
cnt_enter_inzone_light	라이트 인던 입장 횟수
cnt_enter_inzone_skilled	숙련 인던 입장 횟수
cnt_enter_inzone_normal	라이트/숙련 인던 입장 횟수

cnt_enter_raid	레이드 참여 횟수
cnt_enter_raid_light	라이트 레이드 참여 횟수
cnt_enter_bam	밤의 바람 평야 입장 횟수
cnt_clear_inzone_solo	솔로 인던 완료 횟수
cnt_clear_inzone_light	라이트 인던 완료 횟수
cnt_clear_inzone_skilled	숙련 인던 완료 횟수
cnt_clear_inzone_normal	라이트/숙련 인던 완료 횟수
cnt_clear_raid	레이드 완료 횟수
cnt_clear_raid_light	라이트 레이드 완료 횟수
cnt_clear_bam	밤의 바람 평야 완료 횟수
normal_chat	일반 채팅 횟수
whisper_chat	귓속말 채팅 횟수
district_chat	지역 채팅 횟수
party_chat	파티 채팅 횟수
guild_chat	문파 채팅 횟수
faction_chat	세력 채팅 횟수
cnt_use_buffitem	버프 아이템 사용 횟수
gathering_cnt	채집 횟수
making_cnt	제작 횟수

2.2 주제별 Feature Engineering - Activities

"1_1_FE_activity.ipynb" 참조

- train_activity data – data의 마지막 5줄

	wk	acc_id	cnt_dt	play_time	npc_exp	npc_hongmun	quest_exp	quest_hongmun	item_hongmun	game_combat_time	...
440318	5	2b1dd159278e330d1e898f4202c09aa6be22cfb1aac6b7...	7	-0.469571	-0.231874	-0.244973	-0.250423	-0.469541	-0.306354	-0.331656	...
440319	6	2b1dd159278e330d1e898f4202c09aa6be22cfb1aac6b7...	7	-0.461154	-0.231874	-0.182266	-0.250423	-0.441736	-0.277269	-0.334452	...
440320	7	2b1dd159278e330d1e898f4202c09aa6be22cfb1aac6b7...	6	-0.349489	-0.231874	-0.203925	-0.250423	-0.466600	-0.273114	-0.274341	...
440321	8	2b1dd159278e330d1e898f4202c09aa6be22cfb1aac6b7...	3	-0.063085	-0.231874	-0.249229	-0.250423	-0.483695	-0.293889	0.180223	...
440322	8	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	3	-0.650916	-0.231021	-0.290803	-0.245978	-0.487666	-0.306354	-0.567187	...

- Feature Engineering
  - 각 column을 주별로 groupby하여 8주간 총 활동량 변수 생성
  - 각 column을 주별로 pivot하여 8개 변수로 분리해 주별 활동량 변수 생성
  - 주별 활동량의 비율이나 주별 차이를 구해 변수 생성
    - 전체에서 7-8주 or 5-8주의 비율, 7-8주 합과 3-4주 합의 차이
  - NaN값 imputation 방법
    - play_time : play_time의 최소값과 두번째로 작은 값의 차이를 최소값에서 빼준 값 사용
    - 나머지 변수 : 최소값으로 NaN값 사용

2.2 주제별 Feature Engineering - Activities

- 생성한 feature variable: 총 422개

결과 feature		사용 column	feature 설명	data type
play_week_cnt	1개 변수	wk	8주 중에서 acc_id가 play한 week 수	int (1~8)
play_cnt_wk1~ play_cnt_wk8	8개 변수	wk	각 week마다 play한 일 수	int (0~7)
cnt_dt	1개 변수	cnt_dt	8주간 총 play일 수	int (1~56)
play_time	1개 변수	play_time	8주간 총 play time	float
그 외 32개 변수	33개 변수	33개 column	각 column 주별 집계량을 sum으로 groupby한 8주간 총 활동량	float
column별_wk1 ~ wk8	280개 변수	35개 column	각 column을 1-8주별로 분리한 변수	float
column별_7834	25개 변수	25개 column	각 변수의 7,8주 합과 3,4주 합의 차	float
column별 _ratio_78_all	36개 변수	36개 column	각 변수의 7,8주 합 / 1-8주 합	float
column별 _ratio_2half_all	36개 변수	36개 column	각 변수의 하반기(5,6,7,8주) 합 / 1-8주 합	float

2.2 주제별 Feature Engineering - Payment

"1_2_FE_payment.ipynb" 참조

- Payment data: 유저별 주간 결제 금액을 집계 (shape: 800,000 * 3)
- Data schema

payment_week	결제 주 (1-8)	한 id당 8개의 관측치가 나타남
acc_id	결제 유저 아이디	
payment_amount	해당 주 총 결제액	표준화 (거래가 없었던 경우 = 0 → 표준화)

- train_payment data – data의 마지막 5줄

	payment_week	acc_id	payment_amount
799995	4	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	-0.149898
799996	5	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	-0.149898
799997	6	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	-0.149898
799998	7	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	-0.149898
799999	8	7b08dab0401bf3f2ab44bfdfa0b0c2b93d6bed5d8e4350...	-0.149898

2.2 주제별 Feature Engineering - Payment

- 생성한 feature variable: 총 13개

결과 feature		사용 column	feature 설명	data type
pay_amount_w1~ pay_amount_w8	8개 변수	payment_week	각 week별 payment_amount	float
payment_week	1개 변수	payment_week	8주 중 payment가 있었던 week 수	int (1~56)
last_payment_week	1개 변수	payment_week	마지막으로 payment가 있었던 week	float
max_payment_amount	1개 변수	payment_amount	8주간 payment_amount중 최대값	float
sum_payment_amount	1개 변수	payment_amount	8주간 payment_amount의 총 합	float
payment_amount_std	1개 변수	payment_amount	8주간 payment_amount의 표준편차	float

2.2 주제별 Feature Engineering - Party

"1_3_FE_party_network.ipynb"
"1_3_FE_party.ipynb" 참조

- Party Data: 유저간 파티 구성 관계를 집계한 정보 (shape: 6,962,341 * 7)
- Data schema

party_start_week	파티 생성 주 (1~8)
party_start_day	파티 생성 일 (1~7)
party_start_time	파티 생성 시간 (00:00:00.000 ~ 23:59:59:999)
party_end_week	파티 종료 주 (1~8)
party_end_day	파티 종료 일 (1~7)
party_end_time	파티 종료 시간 (00:00:00.000 ~ 23:59:59:999)
party_members_acc_id	파티 구성원 아이디 리스트 파티 참여했던 모든 구성원들의 아이디 기록

- train_party data – data의 마지막 5줄

	party_start_week	party_start_day	party_start_time	party_end_week	party_end_day	party_end_time	party_members_acc_id
6962336	8	6	08:09:30.086	8	6	08:29:51.324	c87c2fad141edf323f3787335b54be22945a02fe052448...
6962337	5	7	11:25:25.719	5	7	11:47:41.557	aafb40d212fe18ff4eafb82fdcf3b53f2161cb3ce59de4...
6962338	7	5	16:29:59.882	7	5	16:30:27.386	86022904c5cf72a54978479c94041f4256d6c3c2a1f71c...
6962339	7	6	23:43:52.265	7	6	23:47:50.285	02181a0c962f34f019bc9d5b582fb0ec79b1441f96aa4d...
6962340	6	5	23:07:31.761	6	5	23:11:01.968	967393e81d99ce8e577ee130b7ce8e4fd45e3e9cecb560...

2.2 주제별 Feature Engineering - Party

- 생성한 feature variable: 총 29개

결과 feature		사용 column	feature 설명	data type
party_cnt	1개 변수	party_members_acc_id	10분 이상 지속한 party에 참여한 총 횟수	int
party_cnt_w1~ party_cnt_w8	8개 변수	party_members_acc_id, party_start_week	각 week별 10분 이상 지속 party에 참여한 횟수	int
party_78_ratio, party_678_ratio	2개 변수	생성한 party_cnt 변수	7-8주 or 6-8주의 party_cnt / 전체 party_cnt_	float (0-1)
party_cnt_std	1개 변수	생성한 party_cnt 변수	각 week별 party 참여 횟수의 표준편차	float
party_total_retained_minute	1개 변수	시간관련 column	참여한 파티의 지속시간 총 합	float
party_total_member_count	1개 변수	party_members_acc_id, party_member_count	참여했던 party의 party member수 총합	int
first, mode, last_party_start(end)_week first, mode, last_party_start(end)_day	12개 변수	party_start_week, party_end_week, party_start_day, party_end_day	최초, 최빈, 최후 party start(end) week, 최초, 최빈, 최후 party start(end) day	int
shortparty_ratio	1개 변수	생성한 party_cnt 변수	10분 미만 지속된 party 참여횟수 / 전체 party 참여횟수	float(0-1)
fix_party_max	1개 변수	party_members_acc_id	한 유저가 특정 유저와 반복해서 party에 참여(고정파티)한 횟수 중 최댓값	float
degree_cent	1개 변수	party_members_acc_id	party network에서 유저의 degree centrality	float

2.2 주제별 Feature Engineering - Guild

"1_4_FE_guild.ipynb" 참조

- Party Data: 문파별 문파원 목록을 집계 (shape: 9,963 * 2)
- Data schema

guild_id	문파 고유 아이디
guild_member_acc_id	문파원 아이디 리스트

- train_guild data – data의 마지막 5줄

	guild_id	guild_member_acc_id
9958	ffe917cf662e746a7491fb55f16151a0f4eff5500b579b...	94eaba795aca5ce53ccbcb8ccf2af788f679f321fd61b3...
9959	ffef2c8316f9aa8e6a29ecd1a0b099bc6f86d4d2048580...	05cd824a467cfc9f8f194c32f735ff52dde8dd898ca532...
9960	ffef446457c4986c7597f680310b347960dfb995a97a1e...	94b8fcf1968f509278e51d7a75e9d22eae0a1d865d9779...
9961	fff2c28b1a1f521eda51809a0568153858dc60b976c291...	293df374edf169385fda206bb90825753186cb0ed65886...
9962	fff5ac3748f18e39c87b31cb52d00b37b4aa205a2b0845...	557f99b8568c83dcf0b6d32f401fbb40b747e1c977b1cb...

- 생성한 feature variable: 총 3개

결과 feature		사용 column	feature 설명	data type
guild_counts	1개 변수	guild_member_acc_id	가입한 길드의 수	int
guild_size	1개 변수	guild_member_acc_id	가입한 길드의 멤버 수 최댓값	int
guild_total_member_count	1개 변수	guild_member_acc_id	가입한 길드의 멤버 수 총합	int

2.2 주제별 Feature Engineering - Trade

"1_2_FE_trade.ipynb" 참조

- Trade data: 유저간 1:1 거래 내역을 집계 (shape: 10,414,351 * 7)
  - train의 10만명 유저 뿐 아니라 다른 유저들의 거래 내역도 포함되어 있음
  - 전체 trade data의 network를 구성하여 중심성 변수 생성
  - 전체 데이터 중 예측 대상 유저의 거래내역을 뽑아내어 변수로 생성
- Data schema

trade_week	거래 발생 주 (1~8)
trade_day	거래 발생 일 (1~7)
trade_time	거래 발생 시간 (00:00:00 ~ 23:59:59)
source_acc_id	주는 계정 아이디
target_acc_id	받는 계정 아이디
item_type	아이템 종류: 6 가지 - money(금), grocery(잡화), weapon(무기), costume(옷), gem(보석), accessory(액세서리)
item_amount	거래된 아이템 수량 (표준화)

2.2 주제별 Feature Engineering - Trade

- 생성한 feature variable: 총 62개

결과 feature		사용 column	feature 설명	data type
sell_cnt / buy_cnt	2개 변수	source_acc_id, target_acc_id	주는 거래를 한 총 횟수	int
sell_cnt_w1~sell_cnt_w8 buy_cnt_w1~buy_cnt_w8	16개 변수	source_acc_id, target_acc_id, trade_week	해당 week에 주는 거래를 한 횟수	int
sell_cnt_d1~sell_cnt_d7 buy_cnt_d1~buy_cnt_d7	14개 변수	source_acc_id, target_acc_id, trade_day	해당 day에 주는 거래/받는 거래를 한 횟수	int
item type별 sell_cnt	6개 변수	source_acc_id, item_type	아이템 타입별 주는 거래를 한 횟수	int
item type별 buy_cnt	6개 변수	target_acc_id, item_type	아이템 타입별 받는 거래를 한 횟수	int
item type별 sell_amount	6개 변수	source_acc_id, item_type, item_amount	아이템 타입별 주는 거래의 거래량 총합	float
item type별 buy_amount	6개 변수	target_acc_id, item_type, item_amount	아이템 타입별 주는 거래의 거래량 총합	float
sell_1st_week sell_last_week	2개 변수	source_acc_id, trade_week	주는 거래가 있었던 첫주 & 마지막 주	int
buy_1st_week buy_last_week	2개 변수	target_acc_id, trade_week	받는 거래가 있었던 첫주 & 마지막 주	int
indegree_cent outdegree_cent	2개 변수	source_acc_id, target_acc_id	trade network에서의 indegree/outdgree centrality	float

2.2 Feature Engineering 결과 - train_merge

- shape of train_merge: (100000, 531)
- 총 feature variable: 529개
- train_merge describe

	play_wk_cnt	cnt_dt	play_time	play_cnt_1wk	play_cnt_2wk	play_cnt_3wk	play_cnt_4wk	play_cnt_5wk	play_cnt_6wk	play_cnt_7wk	...
count	100000	100000	100000	100000	100000	100000	100000	100000	100000	100000	...
mean	4.40323	19.4657	-0.0247526	1.65063	1.81008	2.21524	2.35921	2.41248	2.14319	2.82452	...
std	2.78764	17.183	4.48003	2.60447	2.60422	2.73295	2.71144	2.73408	2.62377	2.69865	...
min	1	1	-5.2893	0	0	0	0	0	0	0	...
25%	1	4	-3.15953	0	0	0	0	0	0	0	...
50%	5	14	-0.659758	0	0	0	1	1	1	2	...
75%	7	30	1.09362	3	3	5	5	5	4	5	...
max	8	56	39.3964	7	7	7	7	7	7	7	...
dtype	int64	int64	float64	float64	float64	float64	float64	float64	float64	float64	...

9 rows × 529 columns

- train_merge – data의 마지막 5줄

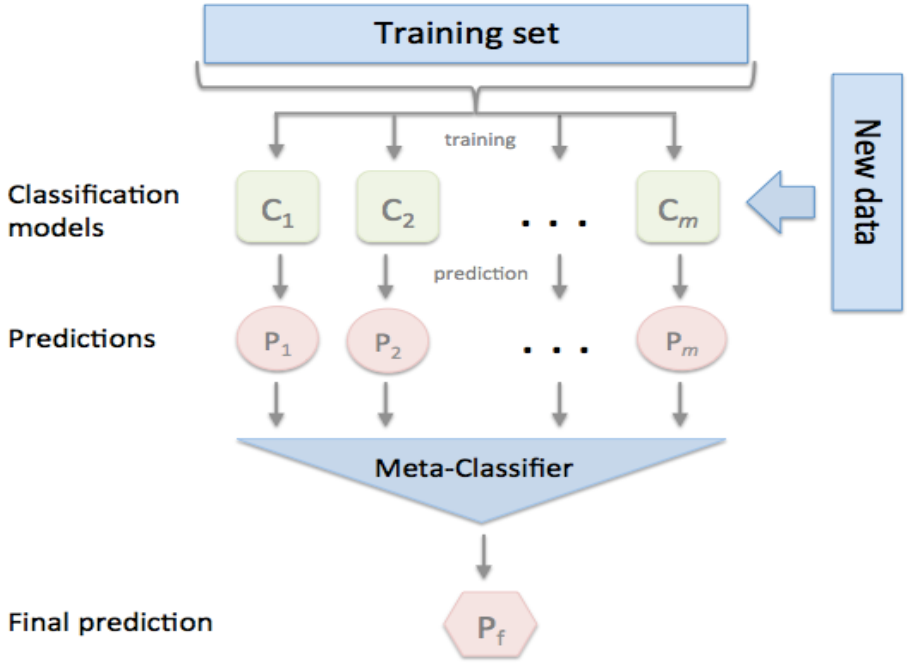
	acc_id	label	play_wk_cnt	cnt_dt	play_time	play_cnt_1wk	play_cnt_2wk	play_cnt_3wk	play_cnt_4wk	play_cnt_5wk	...
99995	da6d33b03968d8e35821f6eb88ad22e12e37aa8867084e...	retained	8	37	-0.065852	2	4	4	6	5	...
99996	676c944f4b6ae63818b3cad824a61233690f16a2275d5d...	retained	6	21	-3.939294	0	1	3	5	3	...
99997	695e1f28e234fc4cc53085e332fa7a76d7895ca4cc745b...	retained	8	47	1.473202	5	7	7	7	5	...
99998	0c87fabaad5542e533f958a1d6fd739993b94e95e00989...	retained	3	8	-1.982065	0	0	0	0	0	...
99999	47ff575cb94019df5695c5d81ec285b0d801607b2a8697...	retained	8	41	2.282317	7	5	3	3	4	...

3. Modeling

3.1 Modeling Overview : Ensemble Technique

Stacked Generalization (Model Stacking) : vecstack패키지 사용*

여러 개의 예측 모델에서 얻은 정보들을 결합하여 새로운 모델을 생성하는 모델 앙상블 기법



*vecstack github: https://github.com/vecxoz/vecstack

사진출처: https://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/#references

3.1 Modeling Overview : Ensemble Technique

- 장점
  - 단일 모델이 아닌 다양한 모델을 활용하여 하나의 모델을 구성
  - 1단계 모델들의 에측 성능이 좋지 않은 부분에 집중하여 좋은 성능을 발휘
  - 1단계 모델의 결과가 현저히 다를수록 효율적임
  - 각 모델별로 다른 독립 변수를 사용 가능함
- 단점
  - 작업량 및 연산량이 많음
  - 예측 성능의 향상이 많지않아 효율성이 떨어짐
  - Train data에 과적합될 가능성이 큼
- 선정 이유
  - 각 클래스별로 분류 성능이 좋은 모델을 활용하기 위해
  - month, 2month에 대해서만 분류 성능이 저하되는 현상을 meta model에서 보완하기 위해

3.2 최종 모델 : 트리 기반 모델의 Stacking

Input : 100000 × 112

Output : 100000 × 3

1st Layer : 3개의 예측 모델 사용

Model	Parameter						
	n_estimators	max_depth	min_samples_split	min_samples_leaf	max_features	bootstrap	learning_rate
RandomForest	294	None	5	2	'sqrt'	False	
Extremely Randomized Trees	1140	None	3	4	'sqrt'	False	
XGBooster	800	7					0.1

2nd Layer (Meta Learner)

Model	Parameter						
	n_estimators	max_depth	min_samples_split	min_samples_leaf	max_features	bootstrap	learning_rate
RandomForest	1452	9	10	4	'sqrt'	True	

4. Model Performance

4.1 Train Data의 평가 Matrix

- 샘플링 방식 : train_test_split(test_size=0.1, shuffle = False)

S_train_data(학습용 데이터) : 90000 × 115

S_val_train(테스트용 데이터) : 10000 ×115

Confusion Matrix

	2month	month	week	retained	합계
2month	1617	328	333	111	2389
month	810	1258	189	243	2527
week	221	81	2147	68	2517
retained	27	119	106	2315	2567

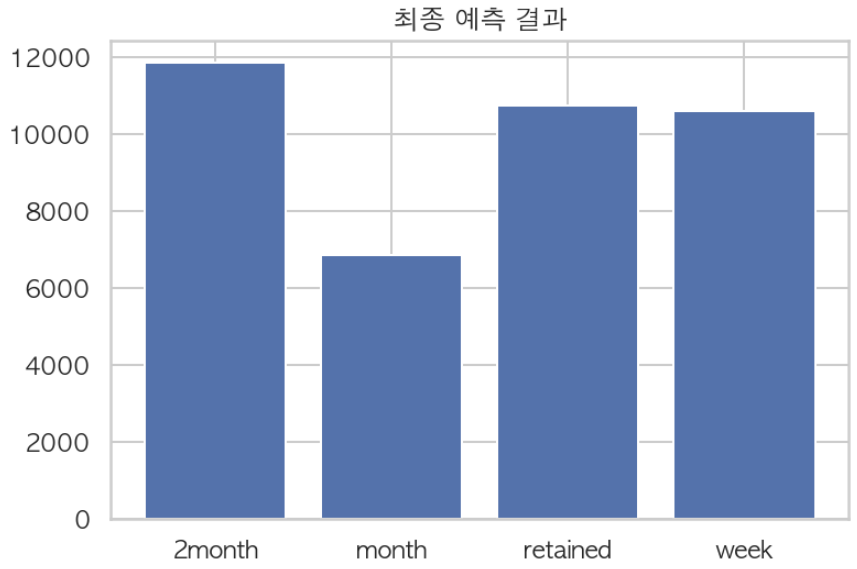
Classification Report

	precision	recall	F1-score	support
2month	0.60	0.68	0.64	2389
month	0.71	0.51	0.69	2527
week	0.77	0.85	0.81	2517
retained	0.85	0.90	0.87	2567
Avg / total	0.74	0.74	0.73	10000

4.2 Test Data의 분류 결과

- 최종 예측값
  - final_pred : 40000 ✕ 2

week	month	2month	retained	합계
10627	6811	11838	10724	40000



5. Conclusion

5. Conclusion

5.1 시사점 및 개선방안

1. 예측 시점 별로 예측 성능에 차이 발생
   - 1주 이내 이탈자와 잔류자는 상대적으로 쉽게 분류된다
     - week (0.87), retained(0.81), 2month(0.64), month(0.59)
   - 4주 이내 이탈자와 8주 이내 이탈자가 유사하게 나타난다.
     - Month의 recall과 2Month의 precision이 떨어지는 경향은 Month가 2Month로 분류되고 있음을 시사한다.

5.2 분석 한계점

⇨ 임의로 나눈 month와 2month를 구분하기 위한 변수를 찾아내는 것이 쉽지 않다

⇨ Stacking을 사용하면 각 모델별로 하이퍼 파라미터를 조절하고 모델을 학습시키는 과정이 복잡하고 물리적 한계가 있다

감사합니다