2차 미니프로젝트_분류

게임 데이터를 활용한 생존 분석

김영경

목차

프로젝트 배경 및 목적

게임 데이터를 활용한 생존 분석

리니지 유저의 활동 데이터를 활용하여 이탈 징후를 보이는 고객을 사전에 선별할 수 있을지,
이탈 예측을 위한 효과적인 분류 모델을 알아보고자 한다.

분류 모델을 통해 이탈 징후를 보이는 고객을 사전에 선별할 수 있다면,
추가적인 인센티브를 제공함으로써 잔존 확률을 높이는 효과를 기대할 수 있다.

프로젝트 개요

진행 순서

DAY1 주제 탐색 및 데이터 확보
DAY2~8 데이터 분석
DAY6~9 시각화자료 제작
DAY10~ 수정 보완

사용 툴

Python (vscode)

사용 데이터

엔씨소프트 오픈데이터,
게임유저 잔존가치를 고려한 고객 이탈예측 (2019년 빅콘 제공데이터)
https://danbi-ncsoft.github.io/OpenData/

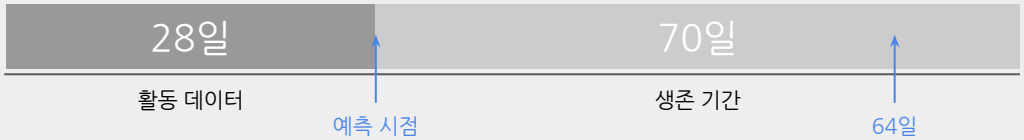
데이터 분석

데이터 설명

데이터 구성

총 4만 건의 예측 시점에서 과거 28일간의 활동 데이터 &
예측 시점 이후 70일간의 관측을 통해 집계된 실제 고객별 이탈 시점(생존 기간) 데이터

*이탈 여부 판단 기간 7일을 감안하여 64일 동안 이탈하지 않은 유저는 잔존으로 처리



데이터셋	데이터 내용
label.csv	대상 유저들의 생존 기간 및 평균 결제 금액
activity.csv	대상 유저의 캐릭터별 활동 이력
combat.csv	대상 유저의 캐릭터별 전투 이력
pledge.csv	대상 유저의 캐릭터별 소속 혈맹 전투 활동 정보
trade.csv	대상 유저의 캐릭터별 거래 이력
payment.csv	대상 유저의 일별 결제 금액

데이터 설명

* 사용하지 않은 정보 삭제, 실제 분석에 사용한 변수만 표기

label.csv

변수	설명
acc_id	유저 아이디
survival_time	생존 기간(일)

payment.csv

변수	설명
day	날짜
acc_id	유저 아이디
amount_spent	결제 금액

activity.csv

변수	설명
day	날짜
acc_id	유저 아이디
playtime	일일 플레이시간
npc_kill	NPC를 죽인 횟수
solo_exp	솔로 사냥 획득 경험치
party_exp	파티 사냥 획득 경험치
quest_exp	퀘스트 획득 경험치
boss_monster	보스 몬스터 타격 여부 (0=미타격 ,1= 타격)

변수	설명
death	캐릭터 사망 횟수
revive	부활 횟수
exp_recovery	경험치 복구 횟수(성당)
fishing	일일 낚시 시간
private_shop	일일 개인상점 운영 시간
game_money_change	일일 아데나 변동량
enchant_count	7레벨 이상 아이템 인첸트 시도 횟수

데이터 설명

pledge.csv

변수	설명
day	날짜
acc_id	유저 아이디
play_char_cnt	게임에 접속한 혈맹원 수
combat_char_cnt	전투에 참여한 혈맹원 수
pledge_combat_cnt	혈맹 간 전투 횟수의 합
random_attacker_cnt	혈맹원 중 막피를 행한 횟수의 합
random_defender_cnt	막피 피해를 받은 횟수의 합
same_pledge_cnt	동일 혈맹원 간 전투 횟수의 합
temp_cnt	혈맹원들의 단발성 전투 횟수의 합
etc_cnt	혈맹원들의 기타 전투 횟수의 합
combat_play_time	전투 캐릭터들의 플레이 시간의 합
non_combat_play_time	非전투 캐릭터 플레이 시간의 합

combat.csv

변수	설명
day	날짜
acc_id	유저 아이디
class	직업
level	레벨
pledge_cnt	혈맹간 전투에 참여한 횟수
random_attacker_cnt	본인이 막피 공격을 행한 횟수
random_defender_cnt	막피 공격을 받은 횟수
temp_cnt	단발성 전투 횟수
same_pledge_cnt	동일 혈맹원 간의 전투 횟수
etc_cn	기타 전투 횟수
num_opponent	전투 상대 캐릭터 수

trade.csv

변수	설명
day	날짜
time	거래 발생 시간
type	거래 구분 (교환창 = 1, 개인상점 = 0)
server	거래 발생 서버
source_acc_id	판매 유저 아이디
target_acc_id	구매 유저 아이디
item_type	아이템 종류
item_amount	거래 아이템 수량
item_price	거래 가격

분석 설계

분류 예측 모델 탐색

‘전체 기간 합산 / 일자별 데이터’ 2종류로 분석 진행 후 학습된 모델 검증

총 8가지 알고리즘 모델로 정확도 점수 파악.

- Logistic Regression (로지스틱회귀분석)
- Decision Tree Classifier (의사결정나무모형)
- Random Forest Classifier (랜덤포레스트)
- Kneighbors Classifier (knn, 최근접이웃회귀모형)
- MLP Classifier (인공신경망)
- SVM
- Keras Classifier (인공신경망)
- ExtraTrees Classifier

추가 정보 탐색

1 무슨 / 몇 개의 캐릭터를 할 때 생존 확률이 높은가?

시간에 따른 그룹별 생존확률 추이를 보기 위해 Kaplan-Meier 모델 사용

2 일별 결제금액이 클수록 생존 확률이 높은가?

시간에 따른 결제금액 그룹별 생존 확률 확인 (Kaplan-Meier)

3 혈맹원 수가 개인의 만족도에 미치는 영향?

혈맹원 수 그룹에 따라 생존 확률 확인

데이터 전처리

종속변수

‘생존 기간(일)’ 컬럼을 이용하여,
63일 이하이면, 0(이탈) / 64일이면, 1(생존)으로 라벨링.

독립변수

전체 기간 합산 데이터의 경우,
데이터.groupby(‘유저아이디’)[‘컬럼명’, ‘컬럼명’,,,,].sum().reset_index()
데이터.merge(데이터, how=’left’, on=’유저아이디’)
데이터.fillna(0, inplace=True)

일자별 데이터의 경우,
데이터.merge(데이터, on= [‘day’,‘유저아이디’], how=’outer’)
데이터.fillna(0, inplace=True)

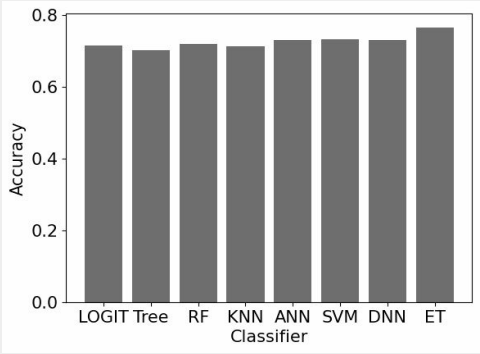
언더샘플링
스케일링

1 분류 예측 모델 탐색 _전체 기간 합산 데이터

정확도 점수

Logistic Regression	0.71534106
Decision Tree	0.7018718
Random Forest	0.71795155
knn	0.71192508
MLP	0.73050429
SVM	0.7308931
Keras	0.72853251
ExtraTrees	0.7649264

f1-score : 0.76



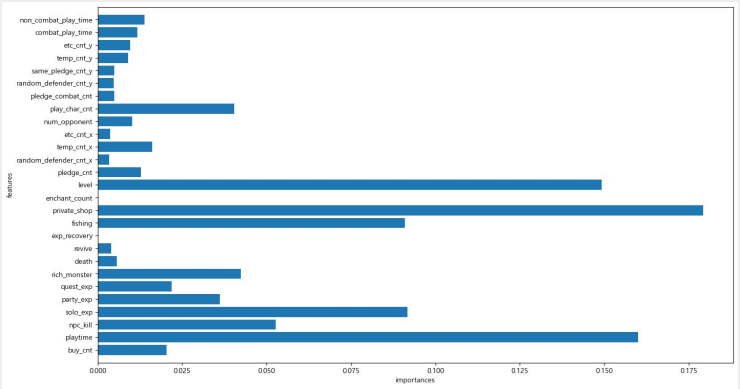
특성 중요도

Decision Tree

playtime, solo_exp, level, party_exp, fishing

Random Forest

private_shop, playtime, level, solo_exp, fishing



1 분류 예측 모델 탐색 _일자별 데이터

정확도 점수

Logistic Regression	0.705128
Decision Tree	0.735107
Random Forest	0.744724
MLP	0.815799
XGB	0.842420
ExtraTrees	0.958029

f1-score : 0.96

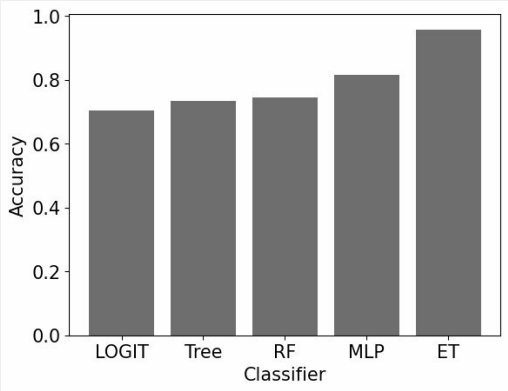
특성 중요도

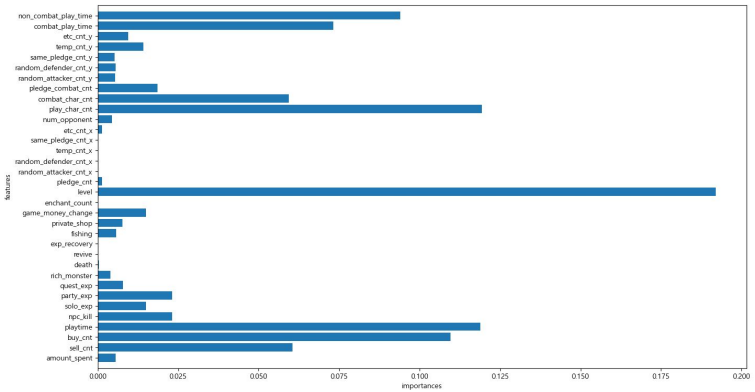
Decision Tree

level, play_char_cnt, buy_cnt, non_combat_play_time, playtime

Random Forest

level, play_char_cnt, playtime, buy_cnt, non_combat_play_time

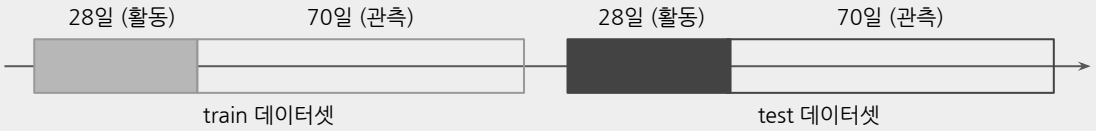




1 분류 예측 모델 탐색 _test 데이터

분석 과정

시점이 다른 2만 건의 test 데이터로 학습 모델의 정확도 확인



정확도 점수

Logistic Regression	0.413601
Decision Tree	0.640903
Random Forest	0.499980
MLP	0.541917
XGB	0.515215
ExtraTrees	0.484261

2 추가 정보 탐색 _무슨 캐릭터를 할 때 생존 확률이 높은가?

분석 과정 시간의 흐름에 따라 알아보고자, Kaplan-Meier Fitter 이용

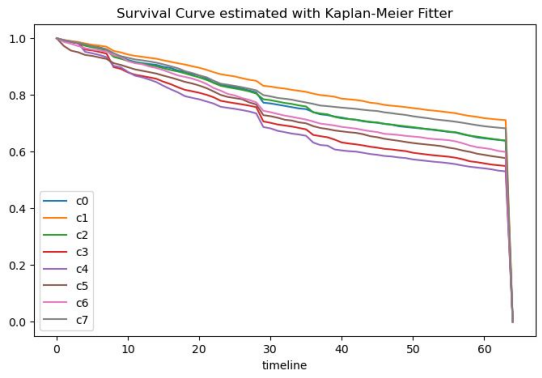
시간 무시하고 단순 숫자로 계산한 결과와 같음 * 한 명이 몇 개의 캐릭터를 갖고 있는지는 고려하지 않음.

결과 캐릭터별 생존율 높은 순서 : 1,7,2,0,6,5,3,4

* 0(군주) 1(기사) 2(요정) 3(마법사) 4(다크엘프) 5(용기사) 6(환술사) 7(전사)

	acc_id	char_id	class	survival_time	survive
0	13809	54861	2	64	1
1	13809	256332	2	64	1
2	13809	307293	3	64	1
3	13809	374964	7	64	1
4	13809	117917	2	64	1
...	...	...	...	...	...
149242	7335	318703	7	39	0
149243	89427	315605	5	25	0
149244	42203	308766	5	31	0
149245	111661	76725	2	64	1
149246	111661	137867	8	64	1

149247 rows × 5 columns



class	survive	cnt	survive/cnt(%)
0	82448	129102	64.0
1	233285	328006	71.0
2	171292	268120	64.0
3	135403	246501	55.0
4	112907	212877	53.0
5	76592	132749	58.0
6	24788	41375	60.0
7	147347	215941	68.0

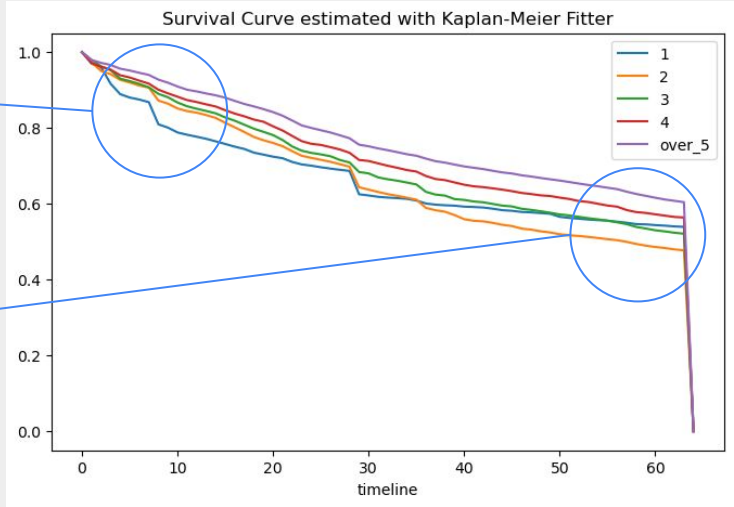
2 추가 정보 탐색 _몇 개의 캐릭터를 할 때 생존 확률이 높은가?

분석 과정 시간의 흐름에 따라 알아보고자, Kaplan-Meier Fitter 이용

결과 생존율 높은 순서 : 5 이상, 4, 1, 3, 2(개)

초반 이탈은 1개 캐릭터 그룹이 가장 빠름

결과적으로는 1개 캐릭터 그룹보다
2,3개 그룹이 더 많이 이탈



2 추가 정보 탐색 _일별 결제금액이 클수록 생존 확률이 높은가?

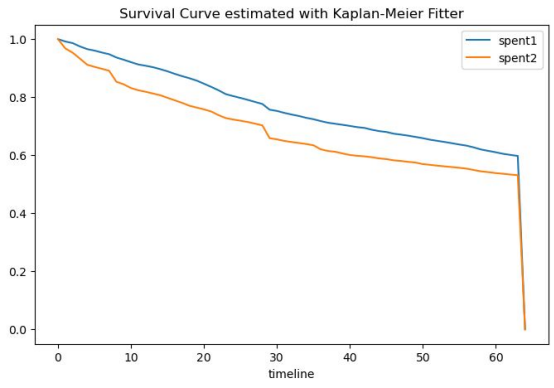
분석 과정 시간의 흐름에 따라 알아보고자, Kaplan-Meier Fitter 이용
시간 무시하고 단순 숫자로 계산한 결과와 같음

결과 생존율 : 평균 이상 결제 그룹(60%) > 평균 이하 결제 그룹(53%)

*추가적으로 6~7일, 28일 부근에 이탈을 유발하는 이벤트가 있었던 것으로 추정

	acc_id	amount_spent	survival_time	survive	spent_label
0	2	0.000000	64	1	2
1	5	0.000000	60	0	2
2	8	1.404644	64	1	1
3	17	0.000000	64	1	2
4	20	0.896531	64	1	2
...	...	...	...	...	...
39995	130463	0.000000	64	1	2
39996	130468	0.000000	1	0	2
39997	130469	0.360255	64	1	2
39998	130470	1.392909	64	1	1
39999	130473	2.955971	64	1	1

40000 rows × 5 columns



spent_label	survive	total	survive/total(%)
0	15326	28839	53.0
1	6670	11161	60.0

2 추가 정보 탐색 _혈맹원 수에 따라 생존 확률이 다를까?

분석 과정

혈맹 활동을 일종의 사회생활이라고 생각하여, 소속 혈맹의 활동 혈맹원 수와 생존 확률 간의 연관성 탐색.
play_char_cnt, combat_char_cnt 두 컬럼에 대해 사분위수로 그룹을 나누어 생존율을 확인했다.

결과

혈맹원들의 활동량과 개인의 생존율은 단순 비례 관계가 아님

```
g0 = df1[(df1['play_char_cnt'] == 0)]
g1 = df1[(df1['play_char_cnt']>0) & (df1['play_char_cnt'] <= 2.670610)]
g2 = df1[(df1['play_char_cnt']>2.670610) & (df1['play_char_cnt'] <= 17.395052)]
g3 = df1[(df1['play_char_cnt']>17.395052) & (df1['play_char_cnt'] <= 48.756670)]
g4 = df1[(df1['play_char_cnt']>48.756670)]
```

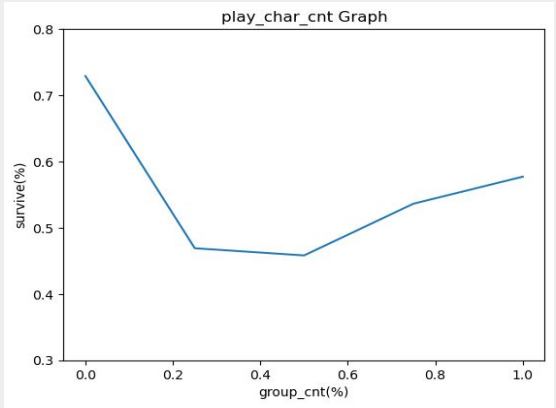
0.0s Python

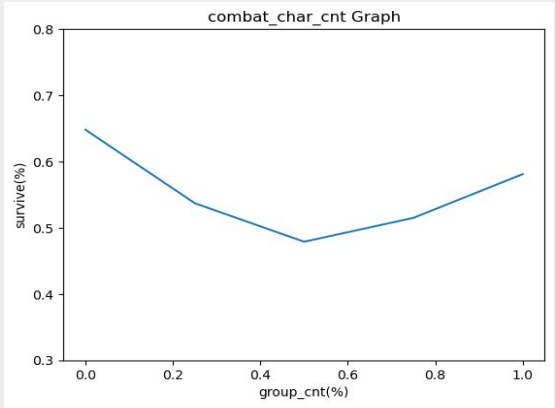
```
G0 = (g0[g0['survive']==1]['acc_id'].count())/(g0['acc_id'].count())
G1 = (g1[g1['survive']==1]['acc_id'].count())/(g1['acc_id'].count())
G2 = (g2[g2['survive']==1]['acc_id'].count())/(g2['acc_id'].count())
G3 = (g3[g3['survive']==1]['acc_id'].count())/(g3['acc_id'].count())
G4 = (g4[g4['survive']==1]['acc_id'].count())/(g4['acc_id'].count())

print('g0 생존율:', G0)
print('g1 생존율:', G1)
print('g2 생존율:', G2)
print('g3 생존율:', G3)
print('g4 생존율:', G4)
```

0.0s Python

```
g0 생존율: 0.7287666775138302
g1 생존율: 0.46887966804979253
g2 생존율: 0.4580335731414868
g3 생존율: 0.5361113893619153
g4 생존율: 0.5768961726791246
```





결론 및 느낀 점

결론

다양한 모델을 활용해 주어진 데이터로 생존/이탈 분류를 시도했고, 생존/이탈 분류의 최종 목표는 생존율을 높이는 것에 있기 때문에 추가적으로 생존에 영향을 주는 요소들은 어떤 것들이 있는지 함께 알아보았다.

분류 모델 중 정확도가 가장 높은 모델은 ExtraTrees Classifier로, 일자별 데이터를 사용했을 때 95.8%의 정확도를 보여줬다. 사용한 데이터가 실제 데이터여서 정상치에서 벗어난 데이터가 많을 것으로 추정되고, ExtraTree는 noise/outlier에 대한 영향을 적게 받는 모델이기 때문으로 추측된다. 다만 다른 시점에 측정된 test 데이터에 적용했을 때는 오히려 점수가 낮은 편인 것을 확인할 수 있었다.

느낀 점

'리니지'라는 낯선 게임에 대한 관련 용어나 문화에 대한 자세히 알아보지 못하고 모델 구현에만 급급했던 점이 아쉽다. 또한, 유저 활동 유형을 나누어서 그룹을 세분화하여 분석했다면 좀 더 정확한 생존/이탈 예측 모델을 구현할 수 있지 않았을까하는 생각도 든다.

추가적으로 생존에 영향을 주는 요소들을 탐색해보며, 게임 데이터가 비록 가상 환경이지만 인간의 일상생활과 비슷한 점이 많다는 생각이 들었고 분석해 볼 수 있는 문제가 다양할 것이라고 생각되었다.

Fin.