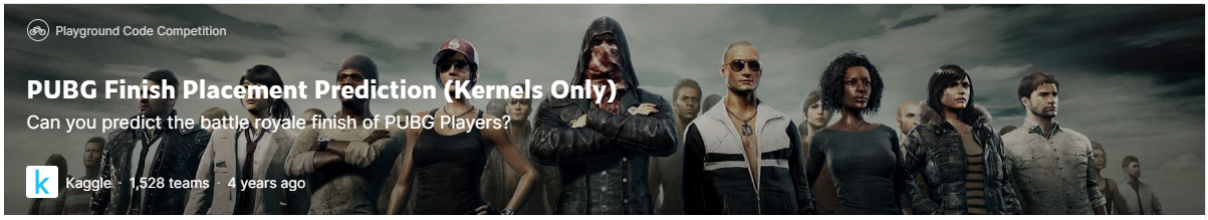
분석 1차 미니프로젝트 중간발표

🔫 승률 예측 및 주요 변수 분석을 통한 유저 레포트 제안 🔫

🌞시각화 20기 이호림& 🌈 분석 20기 강채원/최은선/이민선

1. 주제 & 사용 데이터 소개 및 선정 이유



공통 관심사는 **추천시스템&NLP(자연어처리)**였으나 주어진 시간이 제한적이라 kaggle에서 제공하는 데이터를 활용. kaggle에서 열린 **'배틀그라운드 승률 예측'** 대회의 데이터를 활용하여 **승률 예측 및 주요 변수 분석을 통한 유저 레포트 제안**

👆 **배그 데이터 선정 이유**
1) 게임 데이터를 활용한 분석 프로젝트에 흥미와 관심
2) 배그 마니아

- **원본 데이터 변수들**

Datafield	설명
DBNOs(Down But Not Out)	적을 기절시켰으나, 적 팀원에 의해 부활되어 실제로는 킬로 처리되지 않은 횟수
assists	아군이 킬했을 때 같이 데미지를 넣은 수 (말 그대로 어시스트)
boosts	부스트 아이템 사용 횟수
damageDealt	총 넣은 데미지 (단, 팀에 준 피해나 자해는 포함되지 않음)
headshotKills	헤드샷 킬 수
heals	힐링 아이템 사용 횟수
Id	플레이어 ID
killPlace	매치에서의 킬 횟수 순위
killPoints	킬 수로 산정한 ELO의 개념. 만약 rankPoints에 -1 이외의 값이 있다면, killPoints의 0값은 값이 없는 것으로 처리됨.
killStreaks	짧은 시간동안 최대한 많이 킬했을 때의 킬 수
kills	총 킬 수
longestKill	킬과 킬 사이의 이동한 가장 긴 거리
matchDuration	매치의 시간 (초 단위)
matchId	매치 ID (트레이닝 데이터와 테스트 데이터에 동일한 매치 ID는 없음)
matchType	매치 타입 “solo”, “duo”, “squad”, “solo-fpp”, “duo-fpp”, and “squad-fpp”; 나머지는 이벤트 또는 커스텀 매치
rankPoints	랭킹 ELO의 개념
revives	아군 부활 횟수

rideDistance	탈것으로 이동한 총 거리
roadKills	탈것으로 킬한 횟수
swimDistance	수영한 총 거리
teamKills	팀킬 횟수
vehicleDestroys	차량 폭파 횟수
walkDistance	걸어간 총 거리
weaponsAcquired	얻은 무기의 수
winPoints	승리 횟수로 판단한 ELO의 개념
groupId	매치 내 그룹 아이디, 동일 그룹 구성원들이 다른 게임을 할 때마다 새로운 그룹 아이디가 부여됨
numGroups	게임 내 총 그룹 수 (솔로 -> 듀오 -> 스쿼드로 갈 수록 줄어듦)
maxPlace	매치 내에서 최악의 등수
winPlacePerc	**예측 목표, 1에 가까워질 수록 순위가 높아진다. (0부터 1까지의 값으로 나타냄. 1은 1등을, 0은 꼴등을 의미한다.)**

예측해야할 y 값(종속변수)는 winplaceperc(승률) 변수

2. 현재 진행 상황

- **메모리 용량 축소**

```
for column_name in df:
    if df[column_name].dtype=='float64':
        df[column_name] = pd.to_numeric(df[column_name], downcast= 'float')
    if df[column_name].dtype=='int64':
        df[column_name] = pd.to_numeric(df[column_name],downcast='integer')
```

```
%%time
df.info()

# Memory usages in Bytes
print("Reduced Memory size: ",df.memory_usage(index=True).sum()/(1024*1024), "MB")

# 데이터의 메모리가 상당히 많이 줄었음을 알 수 있음
```

```
memory usage: 339.3+ MB
Reduced Memory size:  339.2767028808594 MB
CPU times: user 31.5 ms, sys: 1.89 ms, total: 33.4 ms
Wall time: 35.8 ms
```

938MB>339MB로 축소

- **결측치 제거**

```
df.isnull().sum()  #결측치는 'winPlacePerc'(승률 변수)에서 존재
```

```
Id                0
groupId           0
matchId           0
assists           0
boosts            0
damageDealt       0
DBNOs             0
headshotKills     0
heals             0
killPlace         0
killPoints        0
kills             0
killStreaks       0
longestKill       0
matchDuration     0
matchType         0
maxPlace          0
numGroups         0
rankPoints        0
revives           0
rideDistance      0
roadKills         0
swimDistance      0
teamKills         0
vehicleDestroys   0
walkDistance      0
weaponsAcquired   0
winPoints         0
winPlacePerc      1
dtype: int64
```

```
# 전체 4446966(약 450만개의) 행에서 결측치 행 1행은 소수
# winPlacePerc(승률) 변수에서 결측치 값이 있는 1 행만 추출: 별 의미 없는 행
# ---> 최종적으로 이 행 제거
```

- **matchType 분류 후 레이블 인코딩 (Label Encoding)**

```
train['matchType'].value_counts()
```

```
squad-fpp           1756186
duo-fpp              996691
squad                626526
solo-fpp             536761
duo                  313591
solo                 181943
normal-squad-fpp      17174
crashfpp               6287
normal-duo-fpp         5489
flaretpp               2505
normal-solo-fpp        1682
flarefpp                718
normal-squad            516
crashtpp                371
normal-solo             326
normal-duo              199
Name: matchType, dtype: int64
```

총 16개의 매치 타입: fpp는 1인칭, 배그에서의 디폴트 값은 3인칭인데 1인칭과 3인칭은 구분하는 것이 별 차이가 없을 것 같아서 묶어줌 —→ solo, duo, squad

crash는 튕김 현상, flare는 배그 측에서 개최한 이벤트, 승률에는 무관해 보여서 묶어줌—> 추후 제거

```
from sklearn.preprocessing import LabelEncoder
```

```
df['matchType'] = df['matchType'].apply(lambda x: 'solo' if ('solo' in x) else 'duo' if ('duo' in x) else 'etc' if ('flare' in x) or ('crash' in x) else 'squad')
encoder = LabelEncoder()
df['matchType'] = encoder.fit_transform(df['matchType'])
df['matchType'].value_counts()
```

```
3    2400402
0    1315970
2     720712
1       9881
Name: matchType, dtype: int64
```

```
print(encoder.inverse_transform([0,1,2,3])) # duo는 0으로 etc는 1로 solo는 2로 squad는 3으로 레이블인코딩 됨
```

```
['duo' 'etc' 'solo' 'squad']
```

- **이상치 제거**

```
현재 train 데이터에는 정상적인 게임에서 나올 수 없는 행들이 존재
* 이동량이 없는데 킬 수가 존재
* 한 매치에서 최대 킬 수가 해당 매치 참여 인원보다 많은 경우
* 차량 탄 거리가 0인데 로드킬이 1이상인 경우
* 총 이동량이 0인데 킬수, 팀킬 수, 헤드샷, 어시스트 횟수, 무기획득 수 등이 1이상인 경우
.
.
.
```

```
(총 20가지 정도 제거해 줌, 코드는 생략)
```

- **파생변수 추가: 변수들을 조합해서 12가지의 파생변수 생성**

파생변수

- totalDistance = walkDistance + rideDistance + swimDistance
- items = heals + boosts + weaponsAcquired
- total_heals = heals + boosts
- teamwork = assists + revives
- headshots_over_kills = headshotKills / kills
- kill/Distance
- hill/Distance
- killPlace_over_maxPlace = killPlace(kill횟수 순위) / maxPlace(최악의 순위)
- walkDistance_over_heals = walkDistance / heals
- workDistance_ove_kills = df_new['walkDistance'] / df_new['kills']
- groupby(match id).count()
- kills/groupby(match id).count()
    - ex) 100명의 매치에서 5명 죽인 능력 vs 10명의 매치에서 5명 죽인 능력

- **matchtype 테이블 분리: solo/duo+squad**

3. 추후 진행 계획

⭐일정 정리

[~ 2/23 데이터 전처리]

- [x] ~~이상치 제거~~
- [x] ~~파생 변수 생성~~
- [x] ~~matchType 테이블 분리~~
- [ ] 시각화 (주요 변수 위주)
- [ ] 데이터 scaling
- [ ] vif 로 변수 선택
- [ ] feature selection 다양한 시도!
- [ ] OLS 유의성 판단 후 최종 변수 선정

[~ 2/26 승률 예측 - 모델링]

- [ ] 모델링
  - [ ] LinearRegression
  - [ ] LogisticRegression
  - [ ] RandomForest
  - [ ] LightGBM
- [ ] 모델 앙상블
- [ ] 하이퍼 파라미터 튜닝 - GridSearch

[인사이트 도출] - 유저 레포트

- [ ] 변수 coef 정리 후 PPT로 만들기

[~ 2/28 발표 준비]

- [ ] 역할 분담
- [ ] PPT 제작
- [ ] 발표

- 배틀그라운드 게임 데이터를 이용하는 프로젝트이니 만큼 추후 팀원들끼리 직접 PC방에 가서 함께 배그를 해 볼 예정

인사이트—→ 유저 맞춤형 레포트 👩

현재 배그 상황: 매치 종료 후 닉네임, 레벨, 킬, 피해량, 어시스트 수만 나옴

보완해서 만들고 싶은 것: 승률 예측 및 승률에 크게 영향을 미치는 원인 변수를 파악해서 **유저 맞춤형 레포트 제작**