온라인 게임 내 Game Bot의 행동 패턴 인식

32181854 박준영

1. Introduction

온라인 게임, 특히 MMORPG(Massive Multiplayer Online Role-Playing Game)에서의 아이템의 현금화는 메타버스, 대체불가토큰(NFT) 열풍에 이어 많은 관심을 받고 있는 키워드이다. 아이템의 현금화는 게임 내에서 상호작용을 통해 재화를 획득하고(이하 '파밍') 이를 게임 외부에서 현금으로 판매하는 행위를 일컫는다. 메타버스와 NFT를 게임 시스템에 적용해 일명 '돈 버는 게임'을 만드는 PTE(Play To Earn) 물결은 최근 많은 게임사들에서 이어지고 있다. 이러한 현상에 대해 새로운 게임 경제를 구축한 것이라는 긍정적 여론도 있는 반면, '바다이야기'와 같이 사행성이 결합되어 사회적 파장을 일으킬 수 있다는 우려의 목소리도 존재한다.[1]

게임 내 재화의 현금 거래가 늘어나면서, 자동으로 아이템을 파밍하는 프로그램인 Game Bot의 이용도 증가하고 있다. 이 프로그램의 사용은 주로 소위 '작업장'이라고 불리는 전문적인 부정 사용자 조직에 의해 이루어진다. Game Bot은 게임 내 경제에 악영향을 미쳐 건전한 게임 문화 형성을 방해할 뿐만 아니라 탈세나 돈세탁 등과 같은 실물경제의 범죄 활동과 연결돼 사회적 문제를 일으킬 수 있다.[2] 또한 보안에도 문제가 발생할 수 있는데, 게임 계정의 도용, 개인정보 해킹 등은 작업장이 많은 게임에서 골머리를 앓고 있는 부분이다.[3]

따라서 본 설계에서는 Game Bot의 로그를 통해 행동 패턴을 인식하고 이를 해결하기 위한 방안을 모색하기로 한다.

2. Background (Related Work)

게임 내 부정 행위를 방지하는 방법은 사전에 부정 행위를 차단하는 방법과 사후 처리 방식이 있다. 사전 차단 방식으로는 감시 소프트웨어 작동, 사용자 네트워크 감시 등이 있다. 이 방법들은 사용자의 불편을 일으키고, 서버 부하를 발생시킬 수 있으며 소프트웨어의 경우, 우회가 가능하다는 단점이 있다. 또한 이러한 방법으로 모든 불법 소프트웨어를 감시할 수 있는 것도 아니다.

사후 처리 방식은 데이터를 기반으로 증거를 획득한 후 이를 이용해 이상 사용자를 추출하는 방식이다. 사용자의 행위를 로그화 해 기록한 데이터를 방탕으로 이상행동을 탐지한다. 이 방식은 기존에 구축된 데이터를 이용하며 사용자 클라이언트 PC나 네트워크에 부하를 주지 않아 사용자의 불편을 최소화하고 특정 소프트웨어에 종속되지 않는다는 장점이 있다.[4]

Game Bot을 탐지하는 기존의 연구는 개인의 행동 패턴 분석 방법과 소셜 행동(상호작용) 분석으로 나누어진다. 개인의 행동 패턴을 분석하는 방법에서는 일반 사용자와 뚜렷하게 구분되는 플레이 스타일을 변수로 분석한다. 움직임이 일정하게 프로그래밍 되어 있을 것이라는 생각에서 사용자의 이동 패턴을 분석한다. 또한 일정한 행위를 반복적으로 수행한다는 특징에서 착안해 사용자의 전투력과 아이템 수집력, 이동성 등을 변수화 하여 행위 빈도를 분석한다. 이 외에도 사용자의 게임 시간과 레벨을 변수로 사용하거나 사용자 행위 사이의 쉬는 시간(idle time)의 존재 여부를 이용해 봇을 탐지한다. 게임 내 자기 유사성(로그의 반복성)을 이용하는 방법론 또한 알려지고 있다.

소셜 행위 관점 분석에서는 Game Bot의 게임 내 소셜 활동에 주목한다. 재화를 얻는 행위에만 집중하고 타인

---

[1] '돈 버는 게임' 열풍…제2의 바다이야기냐, 메타버스 혁신이냐, https://www.hankookilbo.com/News/Read/A2021111209120003504.

[2] E. Digital. Group laundered $38m in virtual curr encies in 18 months. http://www.engagedigital.c om /blog/2008/10/27/group-laundered-38m-in-virt ual-currencies-in-18-months/,2008.

[3] 작업장, 해킹으로 몸살을 앓고 있는 로스트아크, https://steemit.com/hive-196917/@sindoja/5vb3qj.

[4] 우지영, 김휘강.(2017).온라인 게임 내의 부정 행위 탐지 연구 동향.정보보호학회지,27(4),14-21.

과의 대화 로그가 적다는 특징을 이용하는 방법이 있다. 파티 플레이를 할 경우 Game Bot끼리 파티를 맺어 역할을 분담한 후 재화를 얻는 행위를 반복한다. 이를 이용한 방법론이 존재한다.

3. Design & Idea Explanation

(1) Overall Program Design and Function Description

1) dataset collection

- 데이터셋은 고려대학교 대학원 Hacking and Countermeasure Research Lab (HCRLab)의 'Game Bot Detection' 데이터셋[5] [6] 중 'Player actions features.csv'를 재구성하여 사용했다.

- 49739 rows x 31 columns dataset

(두 개의 non-numeric feature을 제외하고 29개의 columns 사용)

```
- columns: 'collect_max_count', 'Sit_ratio', 'Sit_count', 'sit_count_per_day',
        'Exp_get_ratio', 'Exp_get_count', 'exp_get_count_per_day',
        'Item_get_ratio', 'Item_get_count', 'item_get_count_per_day',
        'Money_get_ratio', 'Money_get_count', 'money_get_count_per_day',
        'Abyss_get_ratio', 'Abyss_get_count', 'abyss_get_count_per_day',
        'Exp_repair_count', 'Exp_repair_count_per_day', 'Use_portal_count',
        'Use_portal_count_per_day', 'Killed_bypc_count',
        'Killed_bypc_count_per_day', 'Killed_bynpc_count',
        'Killed_bynpc_count_per_day', 'Teleport_count',
        'Teleport_count_per_day', 'Reborn_count', 'Reborn_count_per_day',
        'Type'],
```

- Label: 마지막 컬럼 Type의 'Human'/'Bot'

-Human: 43489, Bot: 6250

2) System Spec

-M1 silicon mac 8gb

[Reqirements]

-python 3.9.7

-pandas 1.3.4

-scikit-learn 1.0.1

-Seaborn 0.11.2

-numpy 1.19.5

3) feature selection./ extraction

- 'Player action features.csv'(이하 '원본 데이터')를 두 가지 데이터셋인 final_dataset, final_dataset2로 재구성했다. 전자는 선택된 feature에 해당하는 columns를 그대로 가져왔고, 후자는 feature pair의 비율을 계산한 후 재구성했다.

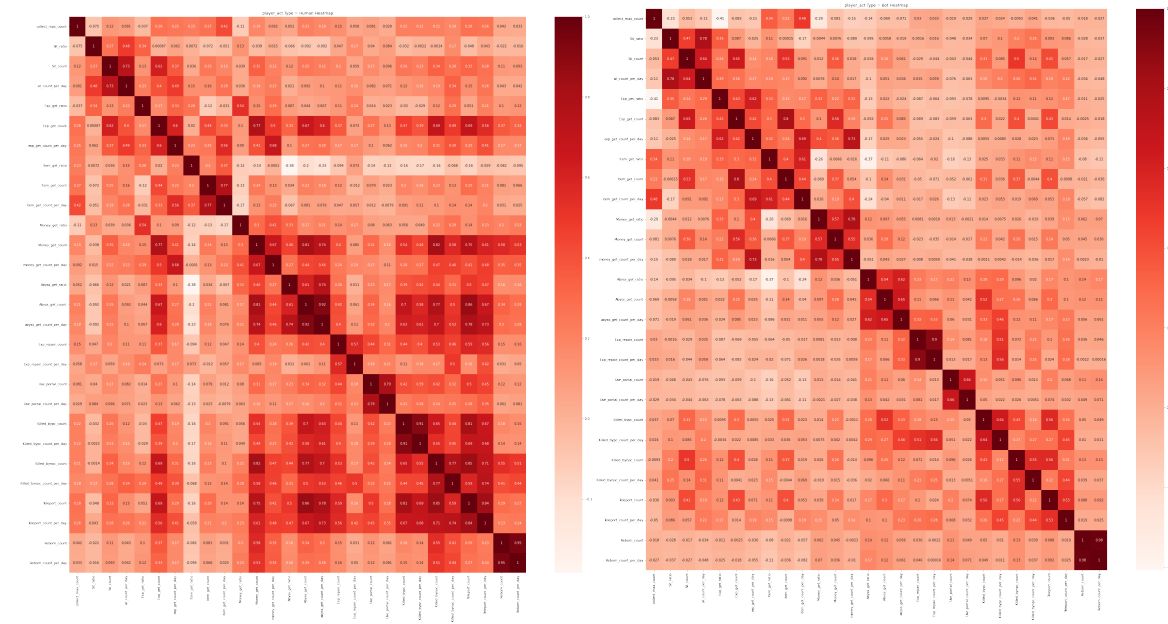
- Feature selection을 위해 데이터의 특징을 확인하고자 원본 데이터를 Type("Human", "Bot")에 따라 나누었다. 분할된 두 데이터프레임에 각각 Pearson Correlation Coefficient (피어슨 상관 계수)를 계산했다. 상관 계수

[5] HCRLab, https://sites.google.com/a/hksecurity.net/ocslab/Datasets/game-bot-detection.

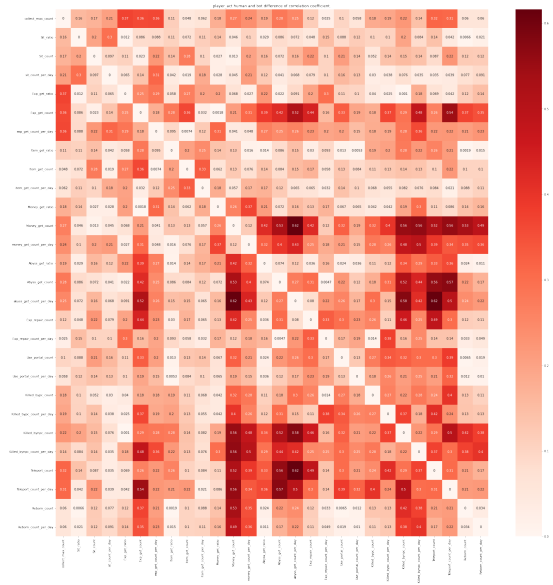
[6] Kang, A. R., Jeong, S. H., Mohaisen, A., & Kim, H. K. (2016). Multimodal game bot detection using user behavioral characteristics. *SpringerPlus*, *5*(1), 1-19.

는 공분산을 평준화한 연관성 측도로, 최소 -1에서 최대 1 사이의 값을 가진다. 상관계수의 절대값이 커질 수록 두 변수가 강한 상관관계를 가지는 것으로 볼 수 있다. 또한 음수 상관계수는 Negative 한 관계를, 양수 상관계수는 Positive한 상관 관계를 의미한다. Figure.1과 Figure.2 는 Human과 Bot의 모든 feature에 대한 상관계수를 히트맵으로 시각화 한 것이다. 두 클래스 사이의 상관계수 차의 절대값이 클수록 클래스를 구분지을 수 있는 feature로 사용 가능할 것이라 생각했다. Figure.3은 두 클래스의 상관계수 차의 절대값을 히트맵으로 시각화 한 그림이다. 상관계수 차의 절대값이 0.5 이상으로 큰 차이를 보이는 feature pair를 선택한 후, 원본 데이터셋의 column을 그대로 가져와 final_dataset을 구성했다. 49739 row x 10 columns (label 포함)이고 포함된 features는 아래와 같다.

```
['abyss_get_count_per_day', 'Exp_get_count', 'Reborn_count',
       'Abyss_get_count', 'Killed_bynpc_count', 'Teleport_count_per_day',
       'Teleport_count', 'Killed_bynpc_count_per_day', 'Money_get_count',
       'Type']
```



(Figure 1.Human Correlation Heatmap /Figure 2.Bot Correlation Heatmap)



(Figure 3.Difference of Human Corr and Bot Corr)

-final_dataset2는 final_dataset과 동일하게 상관계수의 차의 절대값이 0.5 이상인 feature들을 (feature1, feature2) 형태의 tuple로 저장한 후 두 feature 값의 비율을 계산했다. Column의 이름 또한 'feature1/feature2'로 지정했다. 예를 들어 한 샘플의 Teleport_count_per_day 값이 5이고 Exp_get_count의 값이 100 이라면 5/100을 한 후 'Teleport_count_per_day/Exp_get_count' column의 값으로 넣었다. 구성된 columns들은 아래와 같다.

```
'Teleport_count_per_day/Exp_get_count',
       'abyss_get_count_per_day/Money_get_count',
       'Killed_bynpc_count_per_day/Money_get_count',
       'Money_get_count/Abyss_get_count', 'Teleport_count/Abyss_get_count',
       'Exp_get_count/abyss_get_count_per_day',
       'Killed_bynpc_count/abyss_get_count_per_day',
       'Money_get_count/Killed_bynpc_count',
       'Abyss_get_count/Killed_bynpc_count',
       'Teleport_count_per_day/Killed_bynpc_count',
       'Money_get_count/Teleport_count',
       'abyss_get_count_per_day/Teleport_count',
       'Money_get_count/Teleport_count_per_day',
       'Abyss_get_count/Teleport_count_per_day',
       'Money_get_count/Reborn_count', 'Type'
```

(2) classification & Evaluation

- Train data: 45389
- Test data: 4350 (10% of total data)
- StratifiedKFold를 이용해 5-fold validation을 수행, class에 비율에 맞게 fold를 분리하도록 설정

matrics

- confusion matrix
- Linear score
- cross_validation(cv) score
- auc-roc curve
- accuracy
- precision

*** Logistic Regression**

[final_dataset.csv]

- Confusion matrix:

```
 [[7624  210]
 [ 634  485]]
```

- Cv [0.7124771697482649, 0.7026650190025414, 0.7006920730082428, 0.7094950285979867, 0.7033082347905819]
- Mean cv Score 0.7057275050295235
- Linear Regression Coefficient:

```
[[ 5.05882870e-03   6.35862000e-05 -4.00532224e-02 -2.21575365e-04
  -1.39320839e-03 -4.32088146e-02 -2.07418348e-03 -5.48275025e-03
   5.59216483e-05]]
```

- Linear Score: 0.9057299229308612
- prediction of Test dataset: correct=4486

	test_no.	Label	is_correct
0	0	0	True
1	1	0	True
2	2	0	True
3	3	0	True
4	4	0	True
...	...	...	...
4969	4969	0	True
4970	4970	0	False
4971	4971	0	True
4972	4972	0	True
4973	4973	0	True

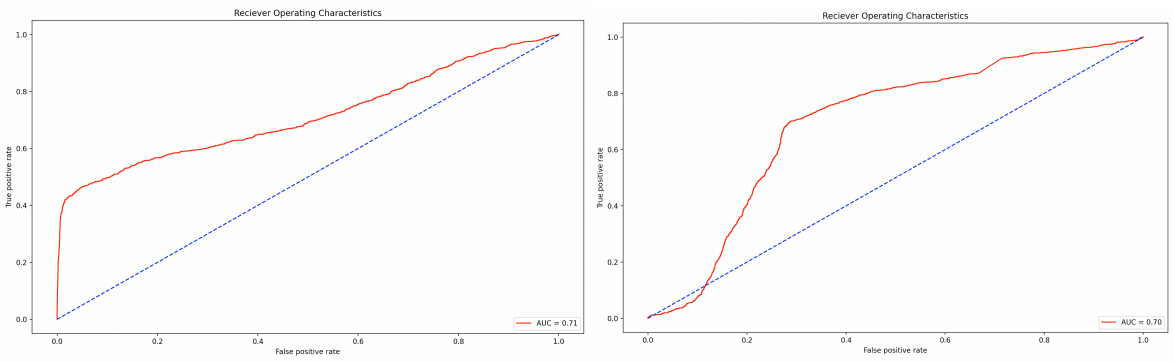
[final_dataset2.csv]
-Confusion matrix"
 [[7834 0]
 [1118 1]]

- Cv: [0.5008658008658009, 0.5008658008658009, 0.5, 0.5, 0.5004468275245755]
- Mean cv Score 0.5004356858512354
- Linear Regression Coefficient:
[[2.06960411e-03 1.13723970e+00 5.73721496e-01 -4.19733031e-04
  4.39941847e-03 -5.83195625e-01 5.50316201e-03 -4.19733031e-04
 -1.73163287e+00 2.06960411e-03 -4.19733031e-04 1.13723970e+00
 -4.19733031e-04 -1.73163287e+00 -4.19733031e-04]]

- Linear Score: 0.8751256562046241
- prediction of Test dataset:

	test_no.	Label	is_correct
0	0	0	True
1	1	0	True
2	2	0	True
3	3	0	True
4	4	0	True
...	...	...	...
4969	4969	0	True
4970	4970	0	False
4971	4971	0	True
4972	4972	0	True
4973	4973	0	True

[4974 rows x 3 columns]
# of corret : 4350



(Figure3.final_dataset roc curve / Figure4.fianl_dataset2 roc curve)

(3) User Interface Design

- 실행 파일은 아래와 같이 구성되어 있다.

```
rawdata_processing.ipynb
Preprocess.py
LogisticRegression.py
> dataset
   final_dataset.csv
   final_dataset2.csv
   (after) Player actions features.csv
   (after) Group activities features.csv
```

- rawdata_processing.ipynb : 원본데이터인 '(after) Player actions features.csv', 'after) Group activities features.csv'를 'final_dataset.csv', 'final_dataset2.csv'로 재구성하는 파이썬 노트북 파일이다.
'after) Group activities features.csv' 본 설계에서는 적합하지 않다고 판단되어 이후에는 사용하지 않는다.
- Preprocess.py : dataframe을 feature로 변환해서 반환한다.
- LogisticRegression.py : Logistic Regression을 수행하고 결과를 반환한다.
-실행 방법은 아래와 같이 LogisticRegression.py를 실행하면 된다. 인자로 csv 데이터셋 주소를 넣어준다.

```
bert-gpu ❱ python LogisticRegression.py --data="./dataset/final_dataset2.csv"
```