Classification of Dota 2 Players

Zhekai Xing, Chi Zhang, Jiayi Li
Electrical Engineering
Columbia University
{zx2198, cz2465, jl4924}@columbia.edu

Abstract— **DOTA 2 is a very popular online game, and each player has his own game style. Although DOTA 2 has a ladder system, it could only show the overall level of the player. This paper applies K-means and Logistic Regression algorithms to classify the players' behavior. K-means is utilized to extract the behavioral characteristics (Labels) of players based on the raw data obtained from DOTA 2 API; Logistic Regression is used to build a series of models which generate labels for a specific player. As result, a web application is built for players to see their labels as well as their similarity with some professional players.**

Keywords-component; DOTA 2; Label; classification; K-means; Logistic Regression; Visualization

I. Introduction

With the development of the Internet, Esports has gradually become a part of the mainstream entertainment. The Esports Economy will grow to $696 million, a year-on-year growth of 41.3%, and the total market will be 1.5 billion by 2020 (Warman). DOTA 2 is one of the most popular Esports programs, which has the highest total awarding prize money ever(Top Games Awarding Prize Money - eSpo...). Same as traditional sports, statistics could help audience watch the game better, it could also help players understand themselves better. Overall, DOTA 2 match is like basketball game, 5 players each side and confront each other. Though DOTA 2 has a ladder system, and players could see their rating scores in the system, it could only reflect the overall level of the players. Also, some website such as Dotamax and Dotabuff provide the detailed statistics of each game, but these trivial data cannot directly reflect the players' characteristics. So, we want to design a system which could generate labels for the players which could directly reflect their behaviors in the game. We think it would benefit both casual and professional players, as well as audiences.

In this project, we try to utilize machine learning methods to analyze game data, and extract the characteristics of each player. More than 30 labels are extracted from our models, and we can generate labels for a specific player with our system.

II. Related Works

Since Esports is a new industry, there is not many related research before. We count all research about DOTA 2 as related work even though some of them have different goal such as prediction of outcome of the games. These research provide us the different ways of data source. There are mainly two methods we can obtains raw data, one is from the DOTA 2 API, another is to analyze the record video.

Another type of research which counts as related work is the classification of traditional sports such as basketball. [1] shows the utilization of K-means to classify NBA players, which is similar to DOTA 2 players.

III. System Overview

Label

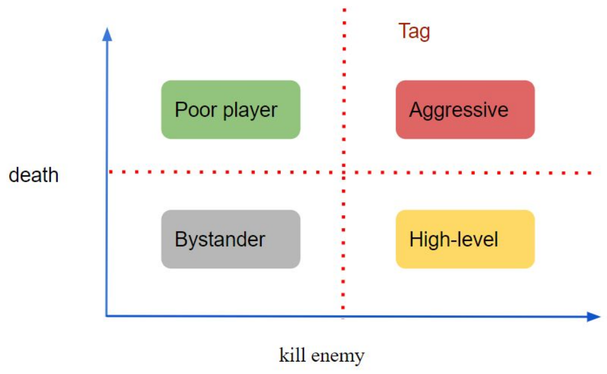


Figure 1. Label schematic

Valve (The company who developed DOTA 2) records every game and provides APIs to developers. There are hundreds of different attributes in the log of a game. Every attribute could reflect an aspect of the player's performance. Labels are some meaningful words that could describe the characteristics of the player. We can generate labels by combination of attributes. For example, In Figure 1, "kill enemy" and "death" are 2 attributes we can get from the API, and the 4 regions represent the 4 labels we can generate from the attributes.

Dataset

We use the DOTA2 API to request raw dataset from the official server, then extract and store useful information into a new .csv file.

The response datasets requested from the server are all in json format. In specific, we first request 100,000 random matches' ids with the command *GET /publicMatches*, using which we can get a list of 100 most recent random public matches once. By recall this command many times we get a file of random ids. Then we use another command *GET /matches/{match_id}* to request detailed information of a single match with a certain match id. Finally we extract useful values from the raw match data and save them into a .csv file for analysis.

Because the official server set a threshold of the amount of request, thus we have to set a time.sleep() function to limit the number of requests in one time. However our request function still crash down and occur internet error in every 20 min. Thus we only efficiently finish reading 10,000 match data, and each match data contain 10 players' information. Furthermore, the content of different match data are also different. Roughly, some matches' data contain any factor we want to use, while other data only cover part of these variables. At last, we extract 50,000 useful players' data.

The raw dataset we extract contain 20 attributes, including: *Account_id, Hero_id, Lose, isRadiant, duration, Gold_per_min, Xp_per_min, Hero_damage, Tower_damage, Stuns, Sentry_uses, Sentry_kills, Hero_healing, Camp_stacked, Kills Deaths Assists, Last_hits Denies Rune_pickup, Courier_kills*

After getting the raw dataset from DOTA2 API, we use R language and Rstudio to clean the raw dataset. Specifically, we let every attribute to divide the attribute *duration*, in order to reduce the influence of time on other factors.

System Architecture

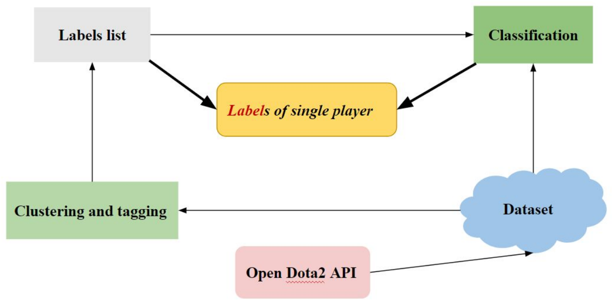


Figure 2. System Architecture

Clustering:

Utilize the valid match info in the Dataset, combine different attributes, use K-means to cluster raw data and generate label list.

Classification:

Prediction models are built based on the result of clustering, these models sit at the backend of the web application. They are responsible for real time generating labels for a user.

Web application:

The web application allows user to query a DOTA 2 id, it will generate the labels for the id. In addition, an interactive graph interface will show the similarity of the id with some famous professional players.

IV. Algorithm

Clustering:

We use the Sklearn.cluster' K-means function in python to cluster different attributes pairs, pair of 2 variables and 3 variables. We draw the distribution plot of different clusters and centers to show the relationship of factors in one pair.
We set 9 centers for each cluster with label 0-8. By analysing the meaning of locations of centers, we can set more meaningful tags for each set of labels.

Classification:

We choose 10 sets of interesting attributes from the dataset and build 10 classification models. Since each sets can generate more than two labels, we use Multinomial Logistic Regression to build the models. It is a classification method that generalizes logistic regression to multiclass problem. The sets we choose are as below:

- Kills & deaths & assists
- last_hits & tower_damage & assist
- Stun & hero_healing & assist,
- Kills & sentry_kills & courier_kills
- Stun & hero_healing & camp_stacked
- Gpm & rune & last_hits
- Gmp & last_hits & kills
- Xpm & gpm
- Hero_damage & tower_damage
- Sentry_uses & sentry_kills

V. Software Package Description

Overview:

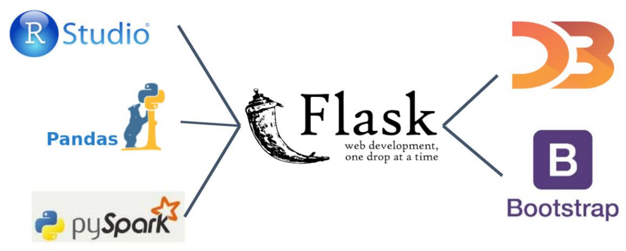


Figure 3. Software tech stack

Clustering & Classification:

Web Application:

We use Flask as backend, and Bootstrap as frontend. All the models are stored in Web/models and will be load into our system when the server boots up.

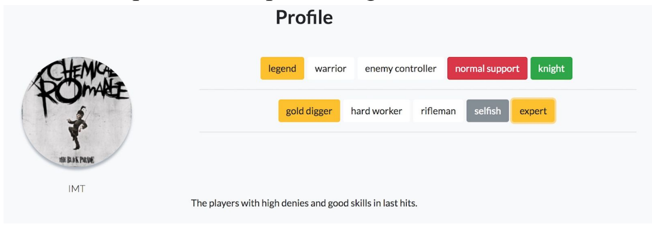
Web/Server.py contains all the functions in the backend. Function query() responds to the user query, the function will retrieve user data and apply the data to the classification models to generate labels. Function graph_json() organizes the data as D3's requirement and reply it to frontend.

As it is time consuming to get user's data from DOTA 2 API as well as applying the data to models, we cache the query history into disk and store it in .Web/obj.

UI:

We use D3 to realize the visualization of data. D3 is a JavaScript library for manipulating documents based on data. D3 helps us bring data to life using HTML, SVG, CSS. D3's emphasis on web standards gives us the full capabilities of modern browsers without tying ourselves to a proprietary framework, without combining powerful visualization components or even without a data-driven approach to DOM manipulation. We mainly use Tree layout and Force-directed graph layout as basic layout of our visualization.

For UI design, there are mainly two parts. One is used to show user's profile and specific tags of that user.



The players with high denies and good skills in last hits.

Figure 4. UI design (Part 1)

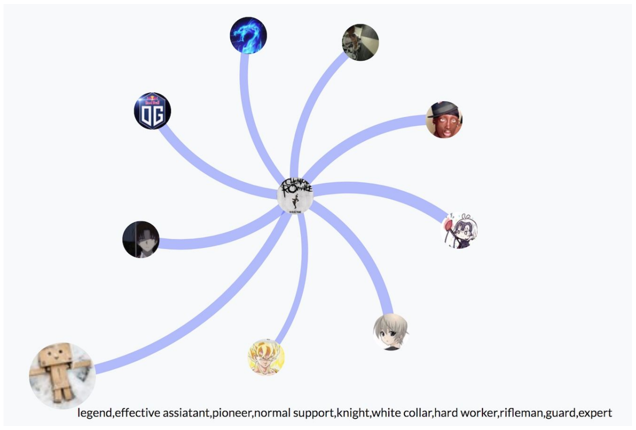
As shown in Figure 4, the player's profile is shown on the left including his username and profile picture. And the buttons on the right are the tags of the selected player, whose colors reflect whether these tags are positive or negative and what its extent is. By clicking these tags, the definition of the selected tags will be shown in the bottom of this part. Some typical definition is shown as follows:

Newbie: The players with extremely low kills and assists but extremely high deaths.

Pioneer: The players with low assists, extremely poor abilities in healing teammates but good abilities in stunning enemies.

Prophet: The players with extremely low kills, poor abilities in killing couriers but excellent skills in cleaning enemies' sentries.

The other part is used to show the visualization of our data.



legend,effective assiatant,pioneer,normal support,knight,white collar,hard worker,rifleman,guard,expert

Figure 5. UI design (Part 2)

As shown in Figure 5, the node in the center of the Figure 5 is the target player. The nodes around him are his friends or some professional players with similar tags, which is selectable by clicking the target user. The width of the links between the target player and professional players are related to the similarity between them. And the text beside the selected user are his tags. By clicking the friends of the target player, we are able to get the topological graph of his friends.

VI. Experiment Results

Clustering:

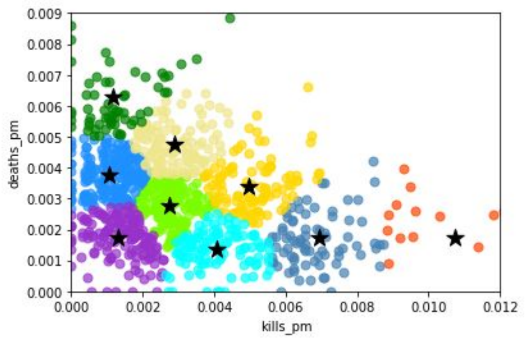


Figure 6. Distribution of pair of 2 variables

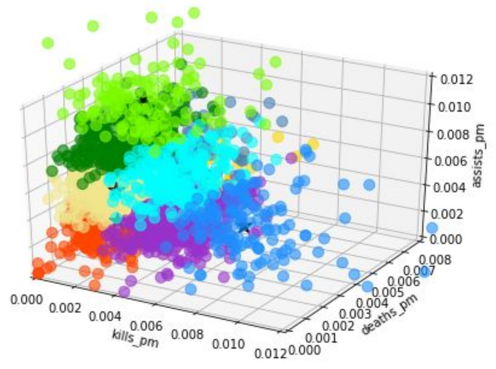


Figure 7. Distribution of pair of 3 variables(1)

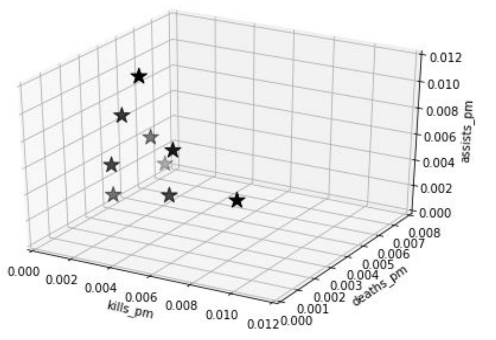


Figure 8. Distribution of pair of 3 variables(2)

Here are two plots of pair of 2 variables and pair of 3 variables.

In the first plot, the x axis is attribute kills_pm, which means the number of enemies killed by players per minute, and the y axis is attribute deaths_pm, which means the time players dead per minutes. Clusters distributed in the whole space along 2 dimensions. Simply, we can define clusters with a high value of kills_pm with a tag of *aggressive*, and define clusters with a high value of deaths_pm with a tag of *weak*.

The second plot has add a z axis of attribute assists_pm, which means the time you help your allies to kill an enemy. Items distribute in the whole space. After excluding colorful points and only show centers of different clusters, we get the distribution of our clusters. And we can also view this 3D plot in different angles, to analyse the relationship of any two factors.

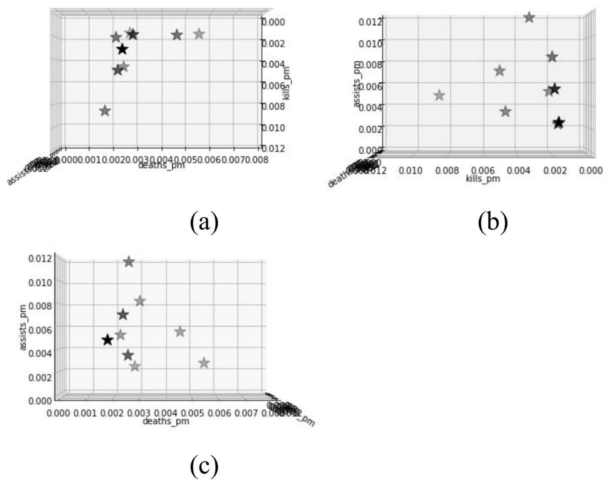


Figure 9. 3D plot from different angles

The image above shows that clusters distribute in 3 dimensions, thus we can read the locations information of each center to translate labels to meaningful tags, like what we do for pair of 2 variables.

Classification:

Model name	Clas s num	precision	Weighted precision
Kills & deaths & assists	4	0.673	0.739
last_hits & tower_damage & assist	5	0.876	0.877
Stun & hero_healing & assist	4	0.949	0.952
Kills & sentry_kills & courier_kills	5	0.366	0.366
Stun & hero_healing & camp_stacked	5	0.873	0.891
Gpm & rune & last_hits	4	0.583	0.590
Gmp & last_hits & kills	4	0.403	0.403
Xpm & gpm	5	0.475	0.518

Hero_damage & tower_damage	5	0.941	0.941
Sentry_uses & sentry_kills	4	0.290	0.314

The precision of the models are as above. 7 out of 10 models have precision more than 0.5. That means the classification models have good prediction of the user behavior and the class number is suitable for the labels.

The left 3 models are Kills & sentry_kills & courier_kills, Gmp & last_hits & kills and Sentry_uses & sentry_kills. We analyzed the raw data of these 3 sets and find in many game sentry_kills are 0 which means most of the players cannot sill sentries in the game. So, the weight of sentry should not evenly distributed in the model or we can just classify this category into yes/no.

VII. Conclusion

This project provides game players a new tool to see their performance in a game. There are many things we can improve next, one thing is we can find more interesting subsets of attributes combination and another thing is that we can optimize data fetching process so that the server can response to user more quickly.

VIII. Appendix

As a team, we discussed the overall This project provides game players a new tool to see their performance in a game. As a team, we discussed the overall architecture of the system. Zhekai Xing mainly focuses on the Classification algorithm and Web backend, Chi Zhang focuses on dataset building and clustering algorithm, Jiayi Li focuses on Web frontend and data visualization.

References

[1] Zhang, L., Lu, F., Liu, A., Guo, P., & Liu, C. (2016). Application of K-Means Clustering Algorithm for Classification of NBA Guards. International Journal of Science and Engineering Applications, 5(1), 1-6

[2] Drachen, A., Yancey, M., Maguire, J., Chu, D., Wang, I. Y., Mahlmann, T., ... & Klabajan, D. (2014, October). Skill-based differences in spatio-temporal team behaviour in defence of the ancients 2 (dota 2). In *Games media entertainment (GEM), 2014 IEEE* (pp. 1-8). IEEE.

[3] Eggert, C., Herrlich, M., Smeddinck, J., & Malaka, R. (2015, September). Classification of player roles in the team-based multi-player game dota 2. In *International Conference on Entertainment Computing* (pp. 112-125). Springer, Cham.

[4] Vandenbroucke, N., Macaire, L., Vieren, C., & Postaire, J. G. (1997, October). Contribution of a color classification to soccer players tracking with snakes. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on* (Vol. 4, pp. 3660-3665). IEEE.