Dota2 Winning Team Prediction from Chosen Heroes by Machine Learning Methods

Final Project of Advanced AI

Jin Haichen

Graduate School of Computer and Information Sciences

Hosei University

Tokyo, Japan

haichen.jin.24@stu.hosei.ac.jp

*Abstract*—This article tries some of the famous machine learning algorithms and neural network to train and predict a Dota2 match winning team from the heroes chosen.

Keywords—dota2 match predict, scikit-learn, mxnet, machine learning, neural network

# Problem description

## Introduction of Dota2

Dota2 is a free-to-play multiplayer online battle arena (MOBA) video game developed and published by Valve Corporation.

Dota2 is played in matches between two teams of five players, with each team occupying and defending their own separate base on the map. Each of the ten players independently controls a powerful character, known as a "hero", who all have unique abilities and differing styles of play. During a match, players collect experience points and items for their heroes in order to successfully battle the opposing team's heroes, who are attempting to do the same to them. A team wins by being the first to destroy a large structure located in the opposing team's base, called the "Ancient". [1]

## Dota2 is popular

Dota2 is a very popular game in the world that more than 100,000,000 users in steam own this game [2] and the peak online player count is 1,291,328. [3]

There are many Dota2 tournaments and the most well-known one is called TI (The International).TI7 (The International 2017) held this year awarded the biggest prize pool in eSports history at over US$24 million, breaking the record set by the previous year’s tournament. The champion won nearly $11 million in prize money. [4]

## My work

A complete Dota2 match can be divided into two parts: heroes choosing and game playing. Both these two parts will bring effect to the match result and there are still no fix strategy on heroes choosing.

My work is focusing on using machine learning methods and neural network to train models by match data before and figure out the heroes choosing influence on matches.

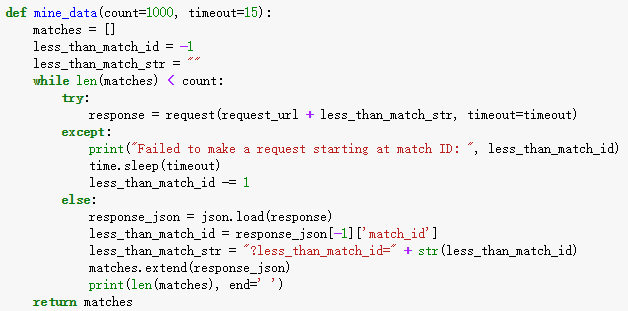
# Data processing

## Get matches data

Valve Corporation does provide APIs for Dota2 matches data grasp but I can get only one match data each access. So I finally decide to use a third party API (opendota API) which can get 100 matches data by one http request. [5]

1. Get matches data





By http request, I can get a json format data block contains the following information.

{"match\_id": 3621587009, "match\_seq\_num": 3148420597, "radiant\_win": false, "start\_time": 1513419640, "duration": 445, "avg\_mmr": null, "num\_mmr": null, "lobby\_type": 0, "game\_mode": 22, "avg\_rank\_tier": 14, "num\_rank\_tier": 2, "cluster": null, "radiant\_team": "95,49,30,6,58", "dire\_team": "28,14,54,84,42"}

I will mainly use the radiant\_win (shows the winning team, true means radiant wins), radiant\_team and dire\_team (list the chosen heroes, the number is the hero id.

In this step, I get 500,000 matches data to train my model.

## Get heroes data

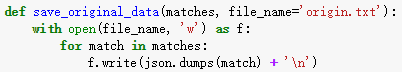
I copy it from a Chinese website and save it in a text file. The format of it is like:“1,敌法师,Anti-Mage”, the first column is hero id, the second and third column is the Chinese and English name of the hero.

It is used in predicting but not in training and testing because the model just use the hero id information as input.

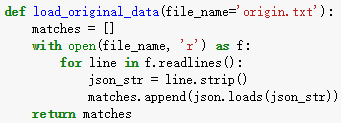
## Manage and organize the data

The dataset is very big, and it costs lots of time to grasp. I save it into file so that I do not have to get it again and again.

1. Write data to text file



1. Read data from text file



For better training effect, I reorganize the heroes choosing data into one-hot format. It is a 120 length list with only 1, 0, -1on it such as:

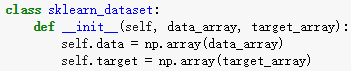
[ 0, 1, 0, 0, -1, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]

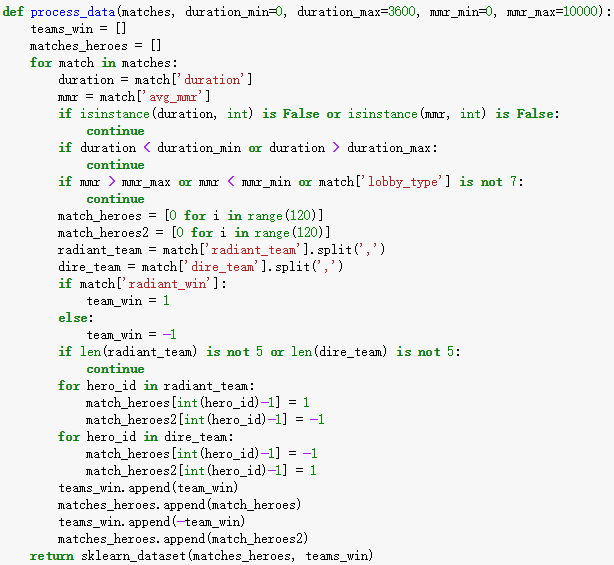
1 means the team 1 chooses the hero with hero id to this index, -1 means the team 2 chooses the hero with hero id to the index and 0 means no team choose them.

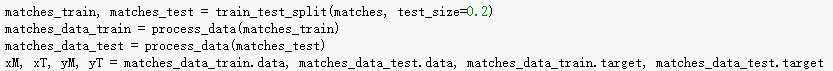
The target 1 means team 1 wins and the target -1 means team 2 wins. And also I decide to treat each match as two data input, one is the winning team as team 1 and one is the opposite. So that it will not only give a positive feedback to the better heroes but also give a negative feedback to the worse heroes.

Then I divide the data for training and testing.

1. Process the heroes choosing data







# Machine learning test result

The training input is the 120 characteristics array and a winning state mentioned before and the testing input is only the characteristics array. The program will predict the winning state.

I have tried several algorithms and here are the result.

## Result with different machine learning methods

As we can see that Gaussian naïve Bayes, Logistic regression and Ada boost have better results than other methods.

The best predict accuracy is around 58%, and no matter how I adjust the parameters, enlarge the dataset, the predict accuracy does not change a lot. I think we can conclude it is the peak level these machine learning methods can reach.

1. Predict accuracy with different algorithms

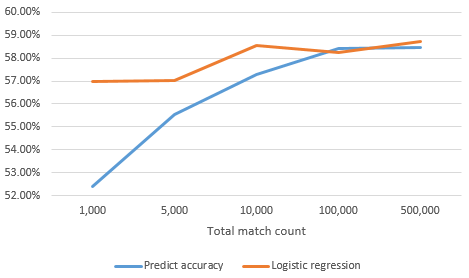
|  |  |
| --- | --- |
| Algorithms | Predict accuracy |
| Gaussian naïve Bayes | 58.45% |
| Perceptron | 52.62% |
| Logistic regression | 58.73% |
| Decision tree | 51.70% |
| Random forest | 54.04% |
| Ada boost | 57.02% |

## Result with different total match count

As we can see in the graph that with more matches data, the model will show better result of predict accuracy. Logistic regression converges much faster than other models.

1. Predict accuracy with different total match count

|  |  |  |
| --- | --- | --- |
| Total match count | Gaussian naïve Bayes | Logistic regression |
| 1,000 | 52.40% | 57.00% |
| 5,000 | 55.56% | 57.02% |
| 10,000 | 57.30% | 58.54% |
| 100,000 | 58.40% | 58.26% |
| 500,000 | 58.45% | 58.73% |



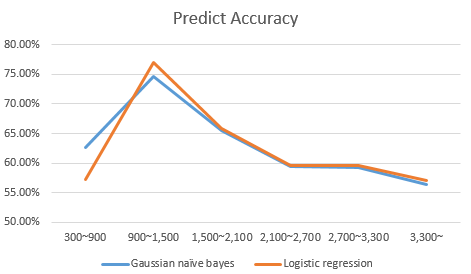
## Result with different duration time

As we can see in the graph that models can predict much more accurate when duration time is low. When duration time is between 900 seconds and 1500 seconds, the predict accuracy reaches the peak: around 75%. We can assume that a group of heroes is more powerful in the early games, so if the match duration time is not too long, they have larger probability to win.

When the match duration time is long, the predict accuracy returns to the normal level. It means when the match does not end fast, then players’ in game strategies and operation may be more important than the heroes choosing.

1. Predict accuracy with different duration time

|  |  |  |
| --- | --- | --- |
| Duration time (s) | Gaussian naïve Bayes | Logistic regression |
| 300~900 | 62.70% | 57.14% |
| 900~1,500 | 74.64% | 77.06% |
| 1,500~2,100 | 65.53% | 65.82% |
| 2,100~2,700 | 59.44% | 59.52% |
| 2,700~3,300 | 59.23% | 59.56% |
| 3,300~ | 56.34% | 57.12% |

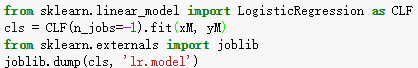


# GUI

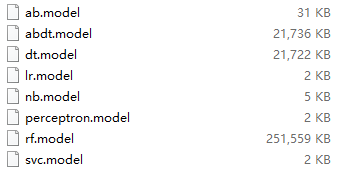
I developed a GUI for the prediction winner and also suggest heroes.

I save all the training models into files and use them to predict winner and suggest heroes.

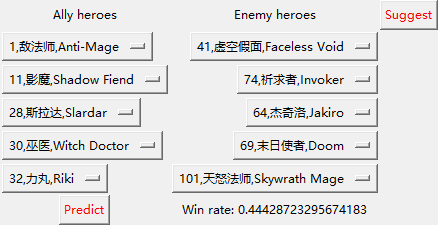
1. Save and Load models





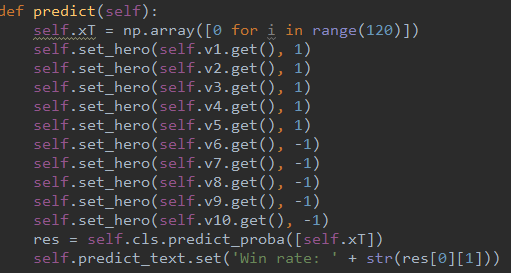


Users choose the heroes each side has chosen and the predict button will show the predicted win rate of the left team. The suggest button will list high win rate heroes according to the heroes each side has chosen.





1. Predict and suggest





I am really sorry that the GUI is a little ugly, but it does work.

The characteristics of logistic regression and naïve Bayes models do not influence each other so that the suggest result will always be same unless the heroes suggested are chosen by either side.

Though I have realized that the logistic regression and naïve Bayes are both not helpful for making suggestions, I still want to finish the suggestion system and see the result by myself.

# Neural network test result

When I tried neural network for the first time, the result is not better than logistic regression and naïve bayes. But when I expanded the dataset to 10 million matches and add more layers to the network, the result became better and better.

## Get matches data

The way to get matches data is almost the same as before, the only difference is that in order to make each single data file not too large, I save them into 20 text files, each of them has 500,000 matches’ data.

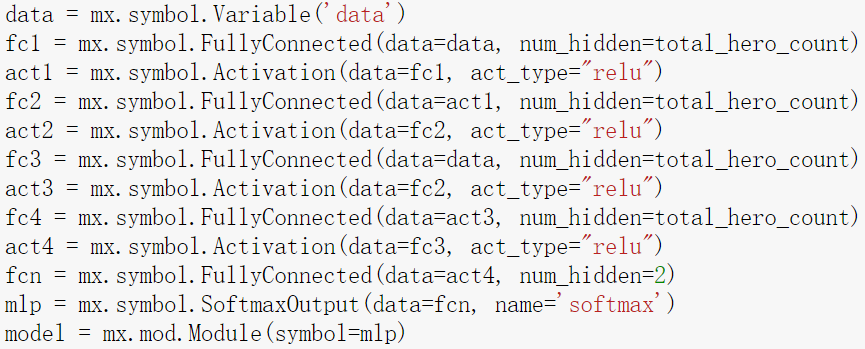
It is really a time consuming process that costs nearly 20 hours to get all the data from internet to local.

Data process is the same as before,

## Build the neural network

I use MxNet as the framework and construct a neural network with 4 hidden layers, each of them has the same count of nodes (hero count). I use fully connected layer and ReLu as the activation, and the input and output is the same as before.

1. Neural network

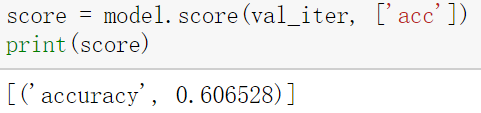


## Trainning and testing

I use the SGD as the optimizer with 0.1 earning rate and 10 epochs. The training is really fast and the predict accuracy of the testing dataset is 60.65%, much better than all the results before.

1. Trainning and testing

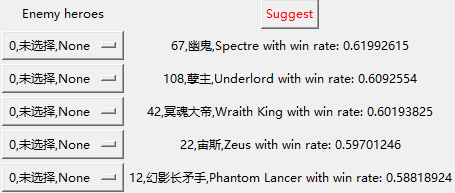


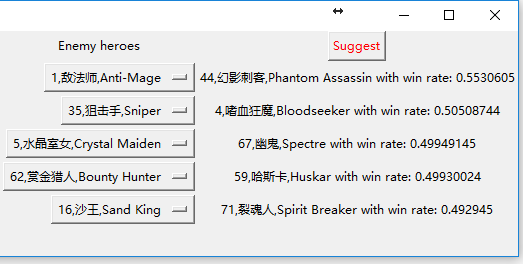


## Predict and suggest GUI

I also save the model for predicting and suggesting GUI. The result is better than all of the machine learning methods mentioned before. The biggest difference also the most important point is that the result shows it can realize the relationship between heroes. It can give better advice due to different situation.

1. Better suggestion





# Conclusion

It is really hard to predict a dota2 match result only by the heroes chosen. When the dataset is small, logistic regression and naïve bayes are good machine learning methods for this type of problem. If we can get more data, then neural network will be a better choice.

Although I took a lot of detours and did not get an ideal result finally, I was still enjoying this project. I have tried many things I wanted to try before but never do, they are all valuable experiences to me.

I am sorry that the report may be awkward in sentence and logic. I spent too much time on coding and finding better methods. As a result I have less time to finish my report and presentation. I think I can do better next time.

# Reference

[1] Wikipedia, Dota 2, https://en.wikipedia.org/wiki/Dota\_2

[2] Steamspy, Dota2, <http://steamspy.com/app/570>

[3] Steamcharts, Dota2, <http://steamcharts.com/app/570>

[4] Wikipedia, The International 2017, <https://en.wikipedia.org/wiki/The_International_2017>

[5] Opendota, API docs, https://docs.opendota.com/