Project 1

Classification of unlabeled LoL match records with “Win/Loss”

1. Overview

League of Legends (LoL) is one of the most played eSports in the world at the moment. Arguably, the fun aspect of games or sport lies within the uncertainty of their outcomes. There is nothing more boring than playing or watching a game, be it soccer, basketball, or any video games, with a predictable ending. This is why La Liga, despite the fact that it used to have two of the best soccer players of the modern time-Lionel Messi and Cristiano Ronaldo-has been widely considered to be less entertaining than her counterpart Premier League in England. The reason is because Barcelona and Real Madrid are usually classes above the rest of the league. To make La Liga more fun to watch, there must be a way to close the skill gaps among teams. Similarly, the creators of LoL have tried their best to match players with teammates and opponents of as similar skill level as possible.

In this project, you will have access to about 3 Million match records of solo gamers as training set. Each record comprises of all publicly available game statistics of a match played by some gamer, including an important field called "winner". If "winner" is “1”, the team 1 won the match, or vice versa.

In sum, the fields include:

- Game ID
- Creation Time (in Epoch format)
- Game Duration (in seconds)
- Season ID
- Winner (1 = team1, 2 = team2)
- First Baron, dragon, tower, blood, inhibitor and Rift Herald (1 = team1, 2 = team2, 0 = none)
- The number of tower, inhibitor, Baron, dragon and Rift Herald kills each team has

Your task is to create one or more classifiers that take as inputs from any fields from above (except “winner) such match record, and labels this record as a "1" or a "2". The test set comprises of ~2 Million of such records.

2. Requirement

The minimum requirement of the project may involve implementing at least one classification method we learn and successfully evaluate the test dataset using the test.py python program. The evaluation result should at least be higher than 50% (i.e. random guess)

You can use Python or the libraries we learnt (Sklearn, Pytorch etc) to realise the implementation.

Should you want to get a distinction, you should use more than one classification methods and try to compare their advantages or disadvantages in terms of this data-set. You may want to compare using the Accuracy or Training time.

3. Report

Your report should include:

- Introduction: A detailed description of the objectives and requirements of the project, and a brief description of the methodology.
- Algorithms. It contains the introduction of the classification methods. The parameters of the algorithm you use should also be introduction.
- Requirements. The prerequisite packages you should use for your code
- Results. A table showing your result and training time, as well as an inclusion of the results of executing your training program captured from the screen.
- Comparison and discussion. Summarize the experience gained in the project. Indicate how your program can be extended and improved if more time is allowed. The documentation for your project is a very important part. The ability of doing a good analysis will also be an important factor to the final assessment of your project, as well as your ability to do proper scientific work.
- It is compulsory to use a word processing tool to write your report. The font size must not be bigger than 12 or smaller than 10. Use 1.5 lines spacing on both sides of a page. Including all diagrams and tables, if any, the length of the report should not be longer than 15 pages.

In addition, all the source files and the report should be uploaded to Github, which is host for software development and version control. If you don't know how you can use Github, you may want to take a look at this:

https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners