Machine Learning Engineer Nanodegree

Capstone Proposal

Parth Mishra
October 26, 2016

Proposal

Predicting Winners in League of Legends

Domain Background

League of Legends[1] in an online multiple player battle arena (MOBA) game that is currently the world's most popular online game with 100 million monthly players worldwide.[2] Like the genre name suggests, League is very much a team game with Role-Playing-Games (RPG) and Real Time Strategy (RTS) influences. There are 10 players total with 5 on each team who are all using a unique in-game avatar known as a "Champion" with various special abilities. Using these Champions, players fight against each other and attempt to take control of each other's respective base and thereby winning the game. Prior to the game actually starting, players on both teams engage in a "draft" where each player selects a champion to use in the game. The choice of champion is very important as a means to best play to the strengths of a particular player as well as select a champion that best fits the overall team composition. There are 137 unique champions in the game, all with their own strengths and weaknesses, thus creating a winning team composition somewhat of an art. In the professional scene, the importance of selecting a good team composition during the draft (also know as "pick-ban phase") is paramount to a team's success. This is due to the fact that at the professional level, individual player skill tends to not vary as much as it does in amateur play. It is still important at lower levels of play since in competitive matchmaking ("ranked") the players are grouped by relative skill.

The problem many players face when entering competitive games is knowing how their team composition will fare against the enemy's team composition. Beyond just the players themselves, coaches, analysts, and spectators often want to have some idea of who is most likely to win a match based on the selected team composition. Assuming relatively similar skill levels, team composition and player mastery present the biggest uncertainties for predicting a team's win or loss. Knowing how likely you are to win with a given team composition and player mastery mitigates this uncertainty to some extent. If a team optimized its picks based on maximizing the overall win rate, then they should logically expect to win more in the

---

[1] This video is a quick, high level introduction to the game: https://youtu.be/BGtROJeMPeE

[2] http://www.forbes.com/sites/insertcoin/2016/09/13/riot-games-reveals-league-of-legends-has-100-million-monthly-players/#7581d79a10b1

long run. Using a supervised learner on previously played games, it would be possible to learn what team compositions are more likely to win.

Part of why I'm submitting a proposal for this topic is because this game is one of my favorites and I've devoted a lot of time and effort into getting better at it myself (I'm in the top 0.02% of player rankings) as well as teaching others how to best maximize their ability to win. I've always wanted to formalize a data centric analytical approach to winning in League of Legends. Riot Games, the maker of the game, does research into this area as well, as evidenced by posts like this one: http://na.leagueoflegends.com/en/page/turbo-mode-data-behind-your-spell-spamming where research was done based on quite literally how often a player presses a certain key per game across all skill levels. From this, insights into the mindset of players based on how often they use or didn't use a certain ability were extracted and backed by real statistical evidence.

Problem Statement

The problem here is that, given perfect information about every champion and player on both teams, which team is most likely to win the game? In order to come up with a reasonable answer to this question, it is possible to use numerous previously played games along with their results to learn over time how well a given team composition will do in a given match. In addition to just team compositions however, player skill/experience can also be factored in using Riot's "Mastery" system which is essentially a separate ranking independent of overall player ranking that numerically (0-7) indicates their mastery with a given champion. Player mastery can be used as a sort of "weight" on individual champion picks that can help in instances where a champion might not be very useful by itself, but if the individual player happens to be very good at using them, it could offset to varying degrees. Note that this is unrelated to a player's win rate on a certain champion. Since this problem regards predicting a binary outcome of winning or losing, this is a supervised learning task. Thus, given a particular game sample that includes every champion picked and their corresponding mastery, it should be possible to predict the winner with reasonable accuracy.

Datasets and Inputs

Due to the niche nature of this problem, there are not very many publicly available data sets to use to tackle this problem. However, through use of the publicly useable Riot Games API[3], I plan to sample ranked game data for players of various skill rankings. Ranked game data refers to the subset of games played that count towards each player's ranking on the server ladder. Ranked data is going to be more indicative of true patterns/results as this the one serious gameplay mode available to the majority of players. The queries of these previously played ranked games return JSON formatted data that can be then transformed into a standard CSV format. Queries on player game data returns numerous categorical data that needs to preprocessed in order to reduce the amount of time needed to process these data in order to get it down to each champion and their respective player's mastery of that champion. As previously mentioned, player mastery is a ranking from 0-7 that indicates how much experience a player has on a champion. This is a more useful metric in this situation compared to win rate since it is independent of

---

3 https://developer.riotgames.com/api/methods

how many times you win or lose. Win rates are often low (< 50%) across the board for players that are ranked lower, thus making it harder to distinguish a winner. Mastery is only achieved by using the particular champion often. Mastery distinguishes itself from pick rate by also having a requirement for occasional good performances on a champion in game prior to advancing mastery level. One possible consideration for its inclusion in this dataset would be to use the average champion mastery level rather than including each individual mastery level. Potential samples could look something like this:

player1_pick	player1_mastery	Outcome
79	4	1

player1	player2	…	player10	team1 mastery	team2 mastery	Outcome
14	54	…	119	4.3	5.8	0

These samples are truncated but would have corresponding values for players 1-10. Every number in a "pick" column refers to one of the champions in the game and can take on the values 0-136 in order to represent all champions currently in the game. Players 1-5 are on one team while player 6-10 are on the opposing team. The outcome label is with respect to players 1-5. The player ordering is random as far as belonging to group 1-5 or 6-10 e.g. a unique player that appears in one ranked game sample as player 2 would be just as likely to appear as player 7 in another ranked game sample so there should be no interference from the player ordering.

As far as sampling the games goes, I will be sampling games from only the top tiers of ranked play (top 10%). The reason is that this avoids situations in which a champion that might be harder to play is a detriment to a team at lower levels of play due to lack of skill whereas in higher tiers, players are more likely to be able to utilize difficult champions and thus the same champion might be a benefit to have on a team. By sectioning off the samples, I can reduce the ambiguity of the resulting dataset. A separate model would be needed for lower tiers of players.

Additionally, one weakness of these data is due to the fact that it's only available for the current iteration of the game known as a "patch". The game gets modified, or "patched", every 2 weeks so it's not guaranteed that the results from the dataset used in the experiment would be necessarily indicative of results a few patches from now. Since we're looking at champion picks and tying them to win prediction, it only makes sense to use data from a single patch as champion abilities and in-game stats change very frequently from patch to patch. In other words, a strong pick in one patch that is likely to predict a winner might get modified the next patch and actually then be a hinderance to their team's chance of victory. Given how often this happens, I plan to only sample from a single patch as there is no point in gathering these data over long periods of time.

Solution Statement

The solution to this problem is going to found by using several supervised learning algorithms, namely:

- SVMs
- Decision Forests
- Logistic Regression

These algorithms give two basic linear classifiers as well as an ensemble method with Random Decision Forests. By learning from thousands of previously played games with knowledge of each champion pick and associated player mastery, these listed classifiers should be able to predict winners with reasonable accuracy beyond just random picking.

Benchmark Model

One benchmark model that could be considered would be using a simple linear classifier such as Naive Bayes to establish a baseline performance. Another possible model could be obtained by looking exclusively at win rates of certain champions and just simply comparing the average win rates of chosen champions by both teams for a given game and whichever is higher, predict that team as the winner. This model would ignore player performance and exclusively model how well a given team composition might do, all else being equal.

Evaluation Metrics

Using a simple accuracy evaluation metric (# of correct classifications / # of total classifications) is a basic but useful metric for determining the usefulness of a binary classifier. This assumes that champions and player skill on their champion vary enough that in a game with all players being close in rank, is enough of an indicator of potential team success. If the accuracy is low for this classifier compared to the benchmark model, this may indicate that player skill for a certain champion is not as important as the champion pick itself or that there's some other underlying relationship that's not accurately being modeled. Given the numerous factors that can influence a game, of which not all are statistical representable e.g. player behavior towards teammates, accuracy is not expected to be super high for any model applied to this problem, but at least enough to be better than random chance.

Project Design

1. Preprocess data

As mentioned before, the data will be collected from the Riot Games API. Given the multitude of data available from any query, only certain parts will be collected, namely: champion picks per player and champion mastery for each player's pick.

2. Data Exploration

Once a satisfactorily sampled dataset is produced, it could potentially be useful to derive some descriptive statistics such as most picked champion, highest win rates, average mastery level, etc. These could be used to contextualize some of the results that we obtain from examining the classifiers performance.

3. Training and Evaluation

In order to see how well the chosen dimensions and features selected for the samples model the data, several implementations of classifiers can be used. Since the data is labeled, this problem calls for a supervised learning algorithm in order to perform a binary classification. No particular supervised classifier stands out as an immediately obvious solution so most likely, several classifiers would be used such as SVMs, Logistic Regression, and a Decision Forest for an ensemble learner.

4. Results Analysis

Once the accuracy results are obtained, comparisons can then be made with the accuracy generated by the baseline model.