Simulating League of Legends Games

Ivan Sheng

Project Overview

- Sports analytics is popular for major sports broadcastings such as the NBA, NFL, and MLB.
- Bootstrap allows for matchups to be simulated thousands of times to gain an understanding of the probability that a player would be struck out, make a free throw shot, win a fight, etc.
- One of the most up and coming industries is competitive video gaming - also known as, esports (electronic sports), which arguably has significantly more statistics and better documented data than traditional sports.
- My project will focus on running monte carlo simulations with bootstrapped probability densities of performance metrics within a professional league of one of the most popular video games in the world, *League of Legend,* to simulate an entire season of outcomes.

Concepts Demonstrated

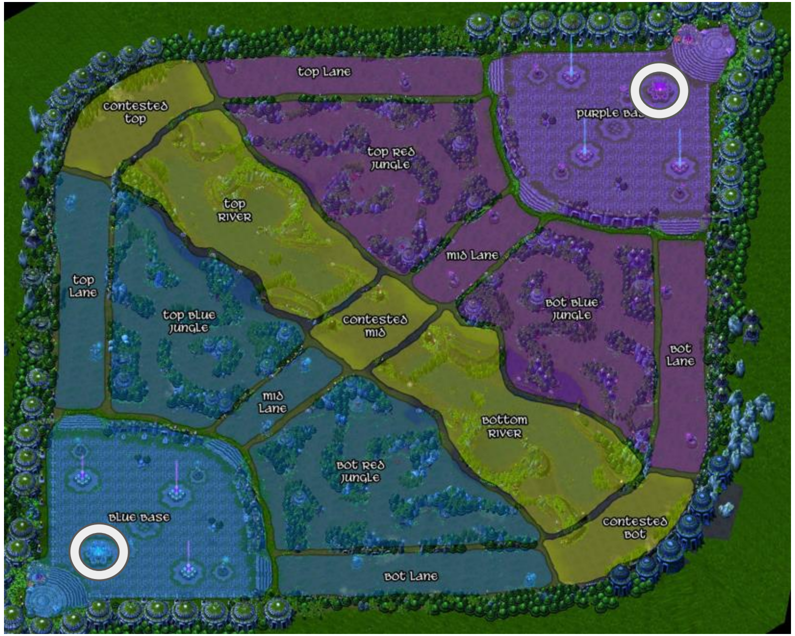
- Taking advantage of bootstrapping, we can develop distributions of how players tend to perform in games.
- Because not all samples (bootstrapped or not) will fit the same distribution, kernel density estimation will be implemented.
- Running thousands of monte carlo simulations, samples will be drawn from these distributions per player in each team matchup, and probabilities of which team will win will be generated based on the results of each simulated regular season and playoffs.

Concept Concerns

- Monte Carlo Simulations
  - Computationally expensive
  - The outputs are only as good as the inputs
- Non-Parametric Bootstrap
  - Straightforward way to derive estimates
  - You wont gain any new knowledge or insight about the true population since sampling from the original population isn’t possible.
  - Assumes that the sample is representative of the population
  - Extremes are removes since this takes the average of the re-samples
- Kernel Density Estimation

Crash Course on *League of Legends*

- League of Legends is a 5 vs 5 game where each player picks one of 5 roles (top, mid, jungle, ad carry, support) and one of over 150 champions (characters) to play.
- It's played on a map where each team has a base, and there are 3 lanes between them with 3 turrets/towers per lane. Each role traditionally belongs to a section of the map, where they largely remain for the first 10 minutes or so.
- The main goal is to destroy the opposing team's nexus (circled in white) by securing objectives (destroying turrets and inhibitors), generating gold (killing enemy champions, neutral monsters, etc), and winning fights.



Crash Course on *League of Legends* eSports

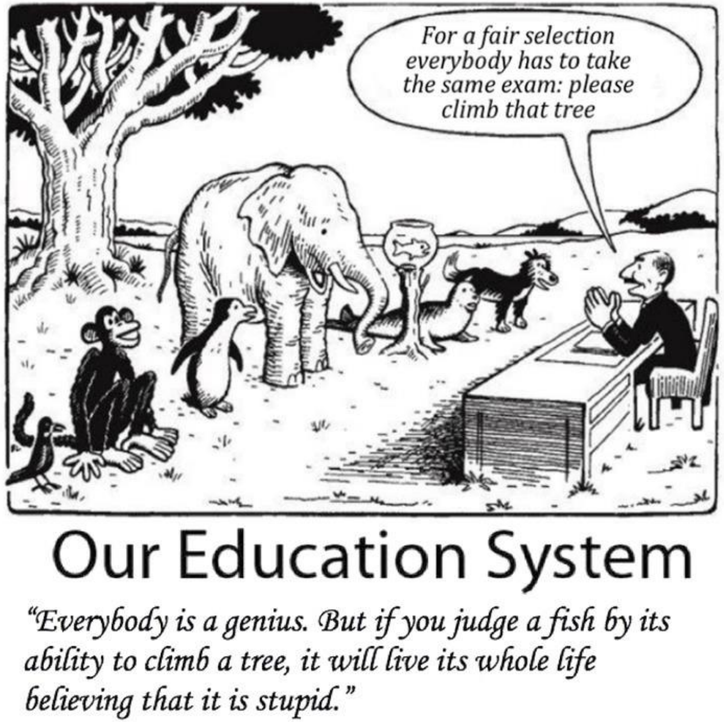
- Most popular online video game with 115 million monthly players and a peak of 50 million daily players.
- 3rd most popular major professional sports league in the United States among 18-34 year olds based on live average minute audience (AMA).
- Events have sold out major stadiums such as Madison Square Garden (NYC), Staples Center (LA), Birds Nest (Beijing), Mercedes-Benz Arena (Berlin), SSE Arena (London), Seoul World Cup Stadium (Seoul), etc.
- 10 teams in the North American LCS (League Championship Series) with a main roster of 5, and a minor-league (academy) team of 5 that can be substituted in and out of.

Data ETL, Bootstrapping, Kernel Densities

1. In-game match statistics for 2019 and 2020 were imported into R
2. Data for the “training” set was filtered for the most recent year of games that don’t include the season to be simulated (Summer 2019 & Spring 2020).
3. *Earned Gold per Minute (Earned GPM)* was the chosen performance metric to determine winning games.
4. If any player had under 36 games (the minimum number of games for a year), the data would be bootstrapped up to that minimum amount.
5. Kernel Density Estimates were calculated for each player based on their *Earned GPM* data set.
6. “Test” set was filtered for the season in question (Summer 2020) in order to create the most recent rosters.

Why Earned GPM?

- Because there are 5 roles in the game, and over 150 champions to choose from, not all role+champions combinations can be measured solely on damage dealt, damage absorbed, health healed, etc. For the most part, gold isn't role dependent.
- Having more gold than other players is a high-level way to tell if a player performed better because faster rates of gold generation require more skill and knowledge.
- Because game time varies based on play-style, performance metrics need to be normalized so longer game times that inherently result in more gold don't equate to better performance.



Monte Carlo Simulation - Round Robin

1. Each player was distributed to their respective teams based on the Summer 2020 dataset. To lower complexity, the starting 5 player for the main roster was dependent on who played the most number of games per role.
2. There are 10 teams in the LCS in total, and each team plays the other 9 teams twice for a total of 18 matches per team.
3. Each opposing team will have random variates drawn from each of their player's kernel density estimates for an earned gold per minute value. The winner is determined by whichever team had the highest total.
4. Running monte carlo simulations, this round robin format was simulated 1,000 times (or 18*1,000 matches in total).
5. A table with the resulting win percentages of each team at the end of all simulations was calculated.

Results: Round Robin

Simulation - Training Set RR 1		Win-Loss	Simulation - Training Set RR 2		Win-Loss	Simulation - 2020 Summer		Win-Loss	Actual - 2020 Summer		Win-Loss	Rank Difference
1	Team Liquid	14-4	1	Cloud9	17-1	1	Cloud9	83.42%	1	Team Liquid	15-3	+1
2	Cloud9	12-6	2	Evil Geniuses	10-8	2	Team Liquid	60.59%	2	Cloud9	13-5	-1
3	Counter Logic Gaming	12-6	3	100 Thieves	10-8	3	Team SoloMid	56.13%	3	FlyQuest	12-6	+4
4	Team SoloMid	10-8	4	FlyQuest	10-8	4	100 Thieves	54.15%	4	Team SoloMid	12-6	-1
5	Dignitas	9-9	5	Team SoloMid	9-9	5	Evil Geniuses	48.09%	5	Gold Guardians	9-9	+1
6	Optic Gaming	8-10	6	Golden Guardians	8-10	6	Golden Guardians	46.46%	6	Evil Geniuses	8-10	-1
7	Golden Guardians	8-10	7	Dignitas	8-10	7	FlyQuest	46.39%	7	100 Thieves	7-11	-3
8	100 Thieves	8-10	8	Immortals	8-10	8	Counter Logic Gaming	40.72%	8	Dignitas	5-13	+1
9	FlyQuest	5-13	9	Team Liquid	7-11	9	Dignitas	33.41%	9	Counter Logic Gaming	5-13	-1
10	Echo Fox	4-14	10	Counter Logic Gaming	3-15	10	Immortals	30.68%	10	Immortals	4-14	0

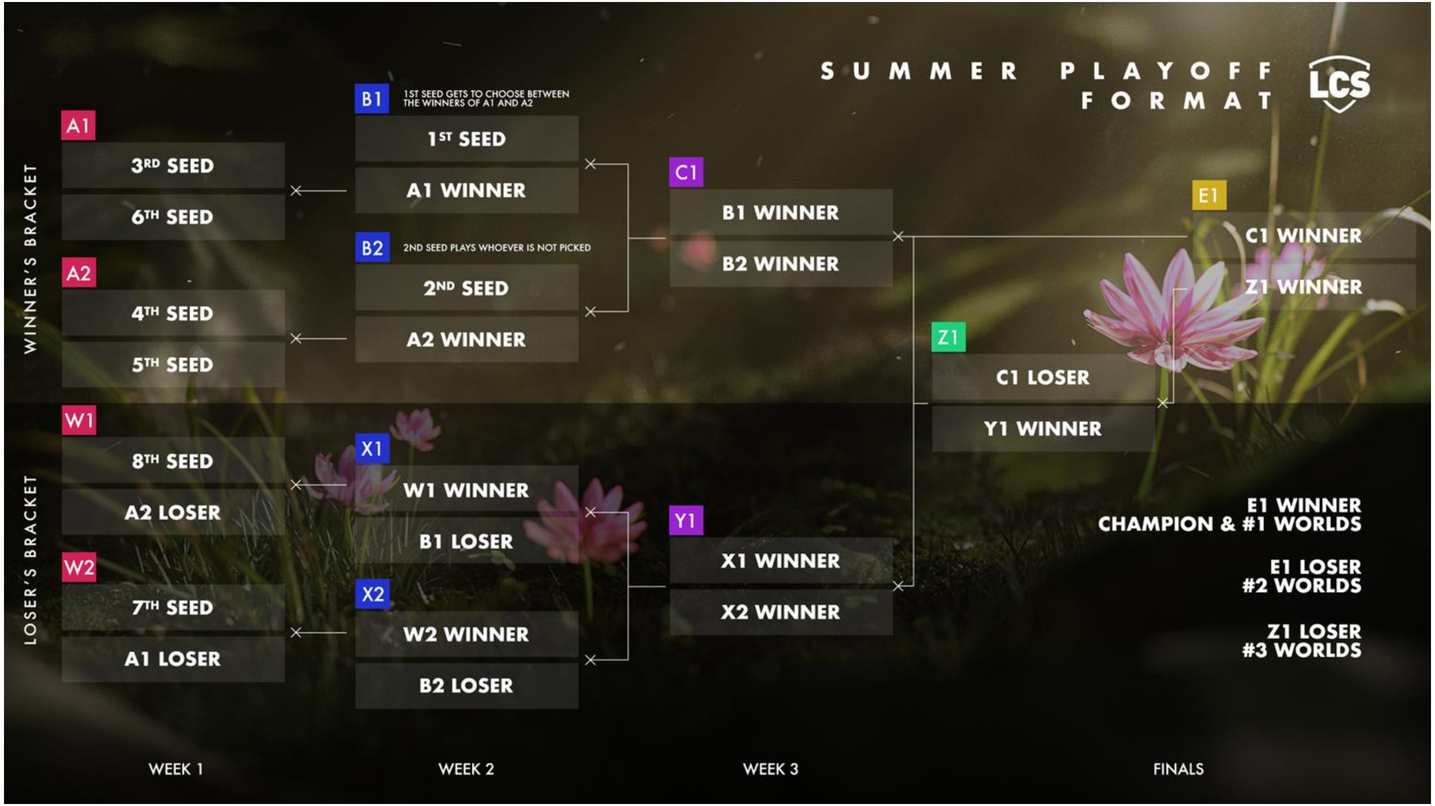
Yellow Highlight: Scores tied, placements were determined via extra-tie breaker matches

Simulation - Training Set Playoffs 1		Win-Loss	Simulation - Training Set Playoffs 2		Win-Loss
1	Team Liquid	6-4	1	Cloud9	9-1
2	Cloud9	5-4	2	FlyQuest	10-9
3	Counter Logic Gaming	7-5	3	Evil Geniuses	5-7
4	Dignitas	7-6	4	Team SoloMid	5-5
5	Team SoloMid	1-3	5	100 Thieves	2-6
6	Optic Gaming	0-3	6	Golden Guardians	0-3

*After 1,000 Monte Carlo simulations of the Round Robin these are the results of the round robin sorted by the final placement ranks.

- Most of the placements were only off by +/- 1 rank.
- Looking at the actual resulting win-loss score, most of these teams were also only a couple wins away from each other during the 2020 Summer season.
- FlyQuest and 100 Thieves seemed to have severely overperformed and underperformed, respectively, in the actual round robin.

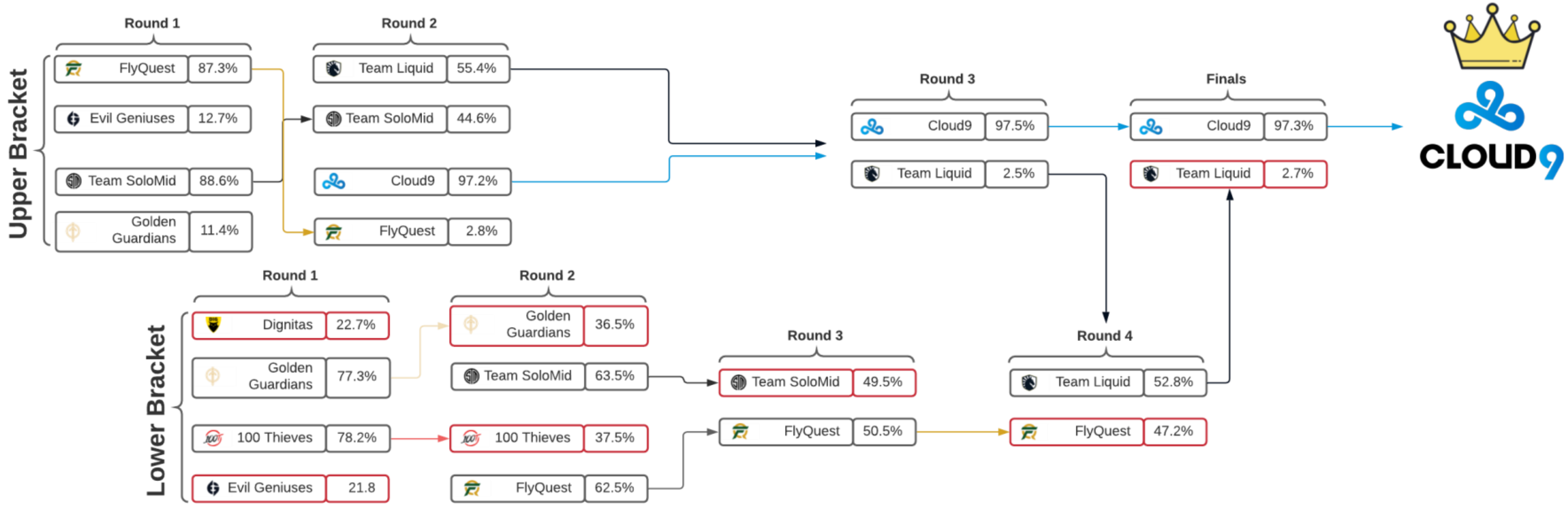
Monte Carlo Simulation - Playoff Bracket Format



Monte Carlo Simulation - Playoff Bracket

1. At this point of the simulation, the actual seeding for the 2020 Summer Playoffs was used instead of the one generated from the simulation.
2. 2019 Summer data was replaced with the actual 2020 Summer Round Robin results, and the kernel density estimations were re-calculated.
3. Each match during playoffs is a best of 5 (meaning the first team to win 3 times out of 5, wins the match), so each Monte Carlo simulation ran until a team acquired 3 total wins.
4. The rankings per simulation was recorded and at the end of all simulations, two win percentages were acquired per match.

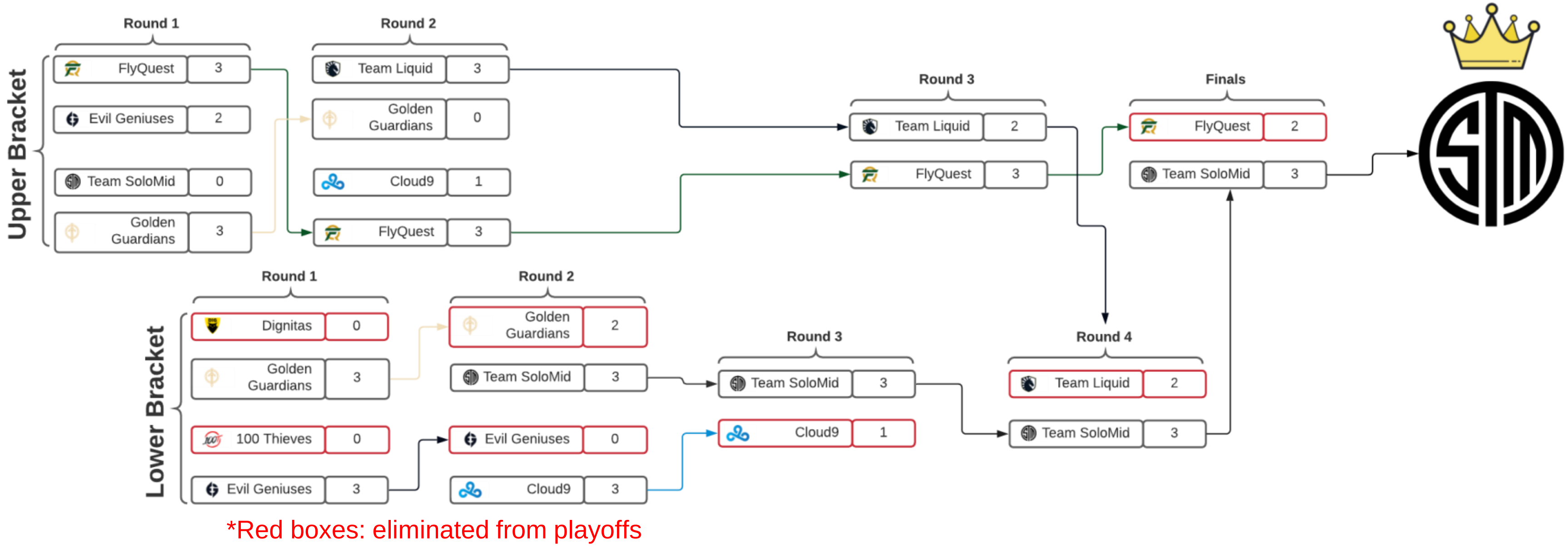
Results: Simulation Playoffs



*Red boxes: eliminated from playoffs

After 1,000 Monte Carlo simulations these are the results of the playoff bracket

Results: Actual Playoffs Results



Results: Playoffs Simulation vs Actual Results

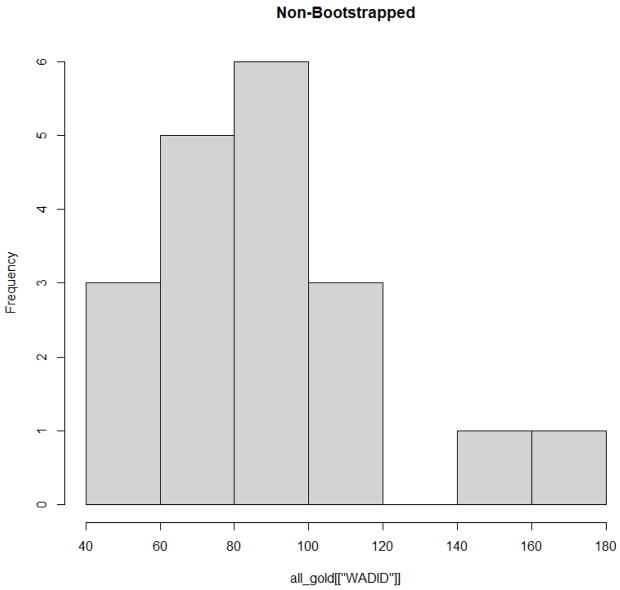
Simulation			LCS 2020 Summer			Rank Difference
1		Cloud9	1		Team SoloMid	+3
2		Team Liquid	2		FlyQuest	+1
3		FlyQuest	3		Team Liquid	-1
4		Team SoloMid	3		Cloud9	-2
5		Golden Guardians	5		Evil Geniuses	+1
5		100 Thieves	5		Golden Guardians	0
6		Evil Geniuses	6		100 Thieves	-1
6		Dignitas	6		Dignitas	0

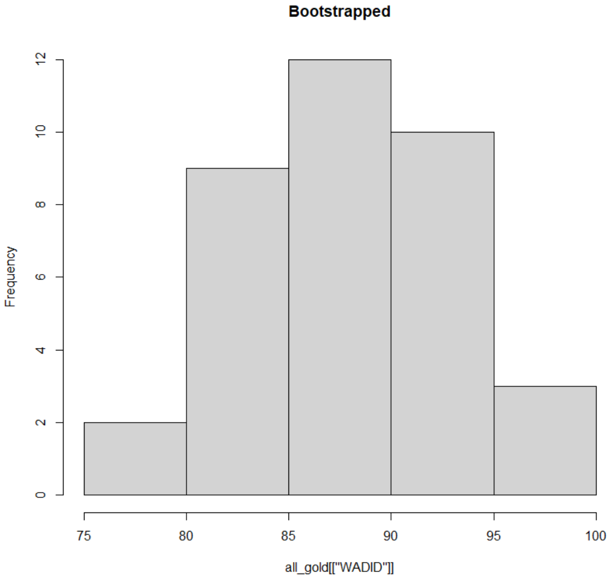
- As seen in the test data, Cloud9 was heavily favored after strong performances, but failed to perform. Looking at their 2020 performance, they won 17/18 of their spring round robin games, 9/10 of their spring playoff games, 9/9 of their first half summer round robin games, but only 4/9 of their second half summer round robin games. This would explain why the simulation placed Cloud9 as the winner so dominantly.
- The games between FlyQuest, Team SoloMid, and Team Liquid were all extremely close in the simulation, and also during the actual playoffs, as well – any of them could've advanced over the other in single simulations.

Appendix

Bootstrap Example

```
hist(all_gold[['WADID']], main = 'Non-Bootstrapped')
kde_results = create_kde(all_gold)
all_gold = kde_results$all_gold
hist(all_gold[['WADID']], main = 'Bootstrapped')
```





Kernel Density Estimate Example

```
p = 'WADID'
bins = seq(min(all_gold[[p]]), max(all_gold[[p]]))
hist(all_gold[[p]], main = 'Earned.GPM')
x = seq(150, 450, length.out = length(player_kernels[[p]]))
plot(x,player_kernels[[p]], type = 'l', main = 'KDE')
```