Competitive League of Legends probabilistic model with Bayesian Networks

Enrico Benedetti
Email: enrico.benedetti5@studio.unibo.it

Abstract. League of Legends (LoL) is a multiplayer online battle arena (MOBA) developed in 2009. It is one of the most played online video games and its eSports scene is one of the largest. In a LoL match, two teams of five players try to defeat the other by destroying the opposing teams' base. While the condition for a win is straightforward, there are many factors that influence the outcome of a game. This work aims to derive a probabilistic model from a dataset of matches played by professional players during the 2021 World Championship. It does not aim to be a model for accurately predicting wins or losses, since it does not account for many variables, such as strategy, coordination or communication of a team, or the psychological state of the players. Nevertheless, it could be useful as a basic tool to allow reasoning upon which of the considered factors contribute to the outcome of a match (and vice-versa).

1 Introduction

The research process for this work can be summarized as follows:

- preprocess the set of competitive game data.
- infer the Bayesian Network's structure from the dataset.
- infer the parameters of the model.
- perform inference on the network using exact/approximate methods.
- discuss results and limitations.

2 Dataset

The dataset is from [1] and contains data from competitive games played at the LoL Worlds 2021, in `.csv` format. The number of games is relatively small, compared to the thousands of matches played every day on each game server, but informative enough if one focuses on the pro players of this particular tournament. The dataset contains many attributes. For the sake of simplicity in structuring the Bayesian Network, some columns have been dropped and some have been combined together. For example 'Kills', 'Deaths' and 'Assists' were combined into a new feature, 'KDA'= $\frac{kills+assists}{deaths}$, which captures most of the information about the three variables.

Furthermore, since most variables are numerical and could potentially assume any value in a very large range, they have been binned to reduce complexity and allow faster testing, although this comes at the price of a loss of information.

Position
Creep Score
Gold Earned
Ward Interactions
Result
KDA
Epic Monsters Advantage

Table 1: Attributes of the dataset after preprocessing. See [2] for a glossary of LoL terms.

The next step is building the Baysian Network.

3 Bayesian Network

A Bayesian Network is a type of probabilistic graphical model. It is a directed acyclic graph, where nodes represent random variables and directed edges represent the probabilistic relationships between them. The presence or absence of edges between nodes indicates dependence, independence or conditional independence between random variables. The whole graph expresses the joint probability distribution for the random variables.

The network structure can be formulated by a domain expert, but when it is difficult to know the causal relationships between the variables,

one can use methods to algorithmically guess those relationships from data.

I chose a mixed approach using a search method from the `pgmpy` library (Hill Climb search), which allowed to apply constraints to the structure and build upon them during search. For instance, I was able to 'hardcode' in the fact that in a match, the player's 'Position' is decided before playing.

Another approach, Tree Search with the Chow and Liu algorithm [3] has also been considered but the result was not satisfying, as it did not seem to capture well the semantics of the game attributes.

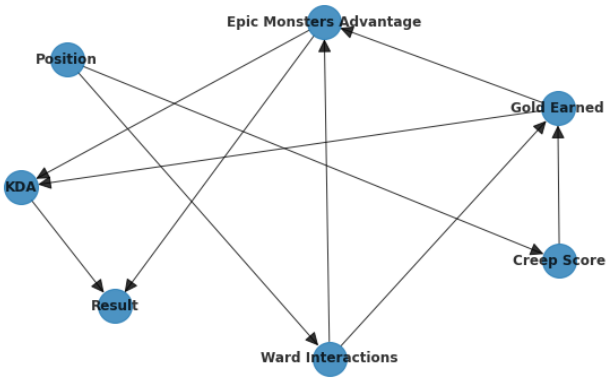

Figure 1: Bayesian Network structure.

To analyze probabilities, the next required step is to provide each node with the conditional probability distribution (CPD) table of that random variable. The network defines a joint distribution of all the variables by a product of the local CPDs:

$$\mathbb{P}(X_1 = x_1, ..., X_n = x_n) = \prod_{i=1}^{n} P(x_i | x_{\text{parents}(i)}).$$

A learning algorithm can be used to learn the parameters of the CPDs from the dataset. The one I used was Maximum Likelihood Estimation (MLE).

$$\max_{\theta} \mathcal{L}(\theta;\ \mathcal{D}) = \max_{\theta} \prod_{x \in \mathcal{D}} \mathbb{P}(X = x;\ \theta)$$

MLE optimizes the parameters θ that maximize the probabilty of the data examples $\mathcal{D}$. The advantages of MLE are that it is a straightforward technique and it can be decomposed into subproblems for each local discribution [4].

4 Inference

The network has all the necessary parameters in place, so we are able to make simple queries (on one query variable) conditioned on evidence variables.

That is, get from the model an answer to 'what is the probability of a certain event?' conditioned on some evidence. As an example, 'what is the distribution of match outcome if KDA is known?':

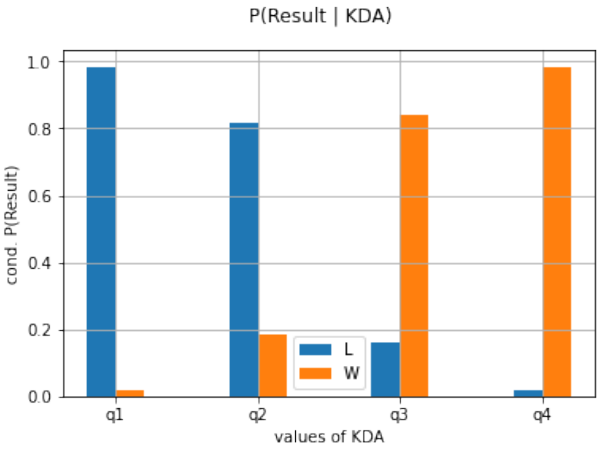

Figure 2: Plot of the above query. It can be observed that KDA is a greatly polarizing attribute for the result.

Mathematically, inference can be described as the following probability (X_q is a set of query variables, X_e are the evidence variables):

$$P(X_q|X_e) = P(X_q, X_e)/P(X_e) = \alpha P(X_q, X_e)$$

Computing this requires considering also the random variables that are not part of the query. The properties of Bayesian Networks allow for computation methods that are better than the exponential cost of enumeration, or one can perform inference using approximate methods based on sampling.

- Exact inference through Variable Elimination. Using the chain rule of probability and finding an appropriate elimination order allows a polynomial time computation [5].

- Approximate inference consists in generating a certain number of samples from a sampling distribution and using those samples for computing the posterior probability. There are many ways to do this. `pgmpy`'s approximate inference class

uses Rejection Sampling and Likelihood Weighting.

Mostly, I have used the functions for exact inference -the variable elimination method-. Because of the reduced size of the model, computation time was not influenced much by the chosen type of inference. Using the `pgmpy` classes for inference, performing a query is very similar but what changes is the underlying methods.

5 Conclusion

The Bayesian Network created for this work produced results that are far for complete in predicting a complex game such as LoL, but nevertheless provided some insight on the relationships between the many aspects of a match. Further work could shift the focus from professional matches only to a greater scope and include game data from all players to get more general information.

Another path be building a model based on games played by a single player or team, to get more accurate predictions. New models could also include a greater variety of data features. In that case, approximate inference should be used for inference if exact methods become too expensive.

References

[1] Rogowski, B. *League of Legends Worlds 2021 Play-In Group Stats*. https://www.kaggle.com/datasets/braydenrogowski/league-of-legends-worlds-2021-playin-group-stats.

[2] Games, P. *League of Legends terminology*. https://primagames.com/featured/lol-slang-popular-league-legends-terms.

[3] Chow, C. and Liu, C. "Approximating discrete probability distributions with dependence trees". In: *IEEE Transactions on Information Theory* 14.3 (1968), pp. 462–467. DOI: 10.1109/TIT.1968.1054142.

[4] Charikar and Sadigh. "Lecture 15: Bayesian networks III". In: *CS221*. Stanford University. 2019, pp. 39–43.

[5] Xing, E. P., Yohan, J., and Baoyu, J. "Lecture 4: Exact Inference". In: *10-708*. 2017, pp. 3–7.