Machine Learning Project Report

Sizhe Fan

Xianglong Wang

Xiaoshuo Yao

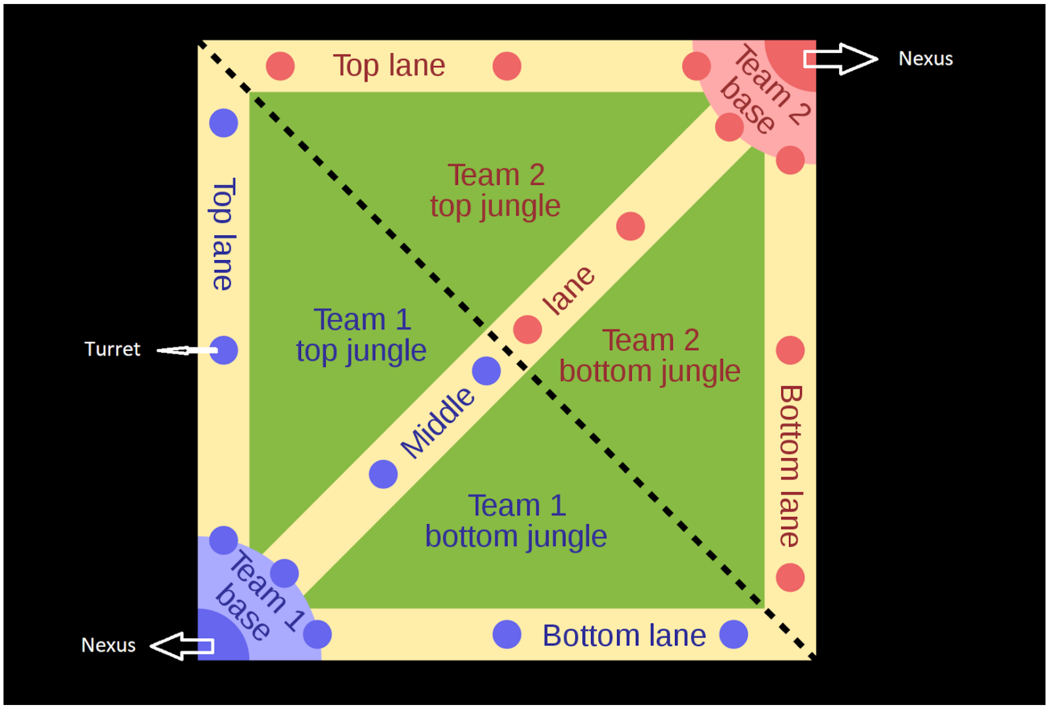
November 6th 2020

Abstract

In this report, we use machine learning and deep learning methods including Regression and Random Forest to make predictions of League of Legends games using the game statistics at 10 minutes and the end of the game. By analyzing these results, we have an insight of how different aspects of the statistics influence the game result. Beyond the game result, we also made a Linear Regression model to estimate the game time using various data that is not seemingly related to it.

1.Introduction & Motivation

League of Legends (LOL) is a Multiplayer Online Battle Arena (MOBA) game. Ten players are divided into two teams (red and blue teams) playing against each other on a map called Summoner's Rift. Each player selects a unique champion with different abilities and acts a specific role in the team. The goal of the game is to destroy the enemy turrets and ultimately, destroy the nexus in the enemy's base. During this process, we can acquire a large amount of data like champion kills, tower destroys, economy difference between the teams and many more.



By Original PNG version by Raizin, SVG rework by Sameboat. - file:Map of MOBA.png (CC 3.0), CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=29443207

It is not a new idea to implement methods of machine learning to the game data. In fact, win prediction based on game data and team statistics have been widely implemented in the E-sport tournaments. The World Championship is the most important League of Legends tournament of the year, there's 16 teams playing in the group stage, then 8 teams face-off in the knock-out matches competing for the trophy. In the 2020 World Championship, a E-sport data analysis organization, Senpai.gg, correctly predicted all 7 games in the knock-out stage by analyzing the data of the teams and applying machine learning methods on the data (Aslan). We were intrigued by their work because they created such a strong algorithm that can predict an event with so much uncertainty. Players in League of Legends games need to make numerous decisions and each of these decisions may influence the result of the game. Also, even the pro player's performance can vary from game to game.

As we were inspired by the prediction of pro games, we started to think of what we can do with the help of machine learning. We expect to increase the understanding of different aspects of the game data and how these data would affect the game. We also want to have a better perception of how computers figuring out the hidden connection between the data. More specifically, can we predict the result of a game based on the early game data? Which statistic affects the game result most? In other words, can the computer properly justify the importance of different game states? Can machine learning help us predict game results other than win-or-lose? And naturally, as a player of the game, what can people do to increase the probability of winning?

In our opinion, machine learning works well for the pro games because the behavior of the pro players is highly unified and logical. However, the competition feature of the game helps us get a reliable dataset. League of Legends have a ranking system and players can only play against players with a similar rank. From low to high, the ranks are: Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster and Challenger. So to make the data more consistent, we used datasets containing only Diamond and above games.

2. Methods

2.1 Data sampling & Data Pre-processing

We collected data from third-party web-sites such as kaggle. The dataset we implied in our model are real gaming datas from the official API of League of Legends.

The dataset is mainly based on high-ranking game matches, which is more consistent in terms of the static. Moreover, all of the datasets contain more than 10,000 samples. Based on the redundancy and consistency of the datasets. We believe these datasets are capable of generating a persuasive regression model.

2.2 Machine Learning Methods and Purpose:

2.2.1 Regression:

Our purpose is to predict game results, such as winning rate or time elapsing, based on data produced during the match. So our goal is to find possible regression relationships between the final result and other gaming datas.

For "continuous" results, such as total enemy killed, time elapsed or wards destroyed, we implement **linear regression** to build a regression model. Linear regression is an effective method of building linear relationships between continuous values. Based on linear regression we found linear relationships between a lot of gaming datas that have no tangible linear relation.

For "discrete" regression, such as the win/lose relationship, we implement **logistic regression** to build a regression model.

Linear Regression:

$$\mathrm{Y} = w_1x_1 + w_2x_2 + w_3x_3 + b$$

Where w_1, w_2, w_3 are weights, b is the bias, Y is the prediction we made for the real label y.

We implied "square-loss" function to detect the difference between prediction "W" and the real label "y" that,

$$l(w_1, w_2, w_3, b) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2}(x_iw_1 + x_iw_1 + x_iw_3 + b - y)^2$$

Logistic Regression:

$$\widehat{Y} = \frac{1}{1+e^{-w^T x}}$$

Where w^T represent weights and the $\widehat{Y}$ represent the probability of a label. The advantage of a logistic regression is that it changes a continuous value to probability corresponded to two labels, which are discrete numbers.

2.2.2 Random Forest:

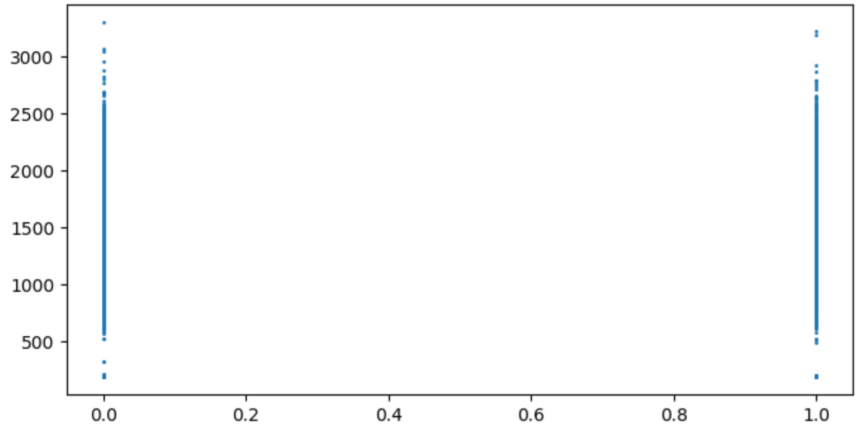
For this part, we want to explore the computer's ability to justify the importance of different game data. Random Forest, included in Sci-kit Learn , is a good way to investigate this topic because it includes the .feature_importances_ method that can give us the importance of each feature of the data (Pedregosa et.al). We want to predict whether the team wins the match or not from both the datasets and estimate the length of the game from the end-game dataset. Both Random Forest Classifier and Random Forest Regressor are included in Sci-kit Learn (Pedregosa et.al). For the game result, a discrete data, we want to use Random Forest Classifier. For the length of the game, a continuous data, we want to use Random Forest Regressor. We use panda(The pandas development team), numpy(Harris et.al) and matplotlib(Hunter) to process the data and draw graphs to visualize the data.

3. Results

3.1 Regression:

We applied linear regression to predict the game duration based on final data and found linear relationships of elements that have no tangible relationships. For example, it is hard to conclude that there exists a linear relationship between game duration and “First blood” or “First dragon”, based on direction observation.

From the picture we know that it is hard to find linear relationships for the features:



However, surprisingly, the model showed a linear relationship between different elements that seems to have no contribution to calculating game duration. The accuracy of the prediction is surprisingly high that the average loss is 0.0116778 (best score).

The linear model we found for game duration and blueWins, blueFirstBlood, blueFirstTower, ...blueChampionDamageDealt(the first 16 column of dataset “GrandMaster_Ranked_Games.csv”):

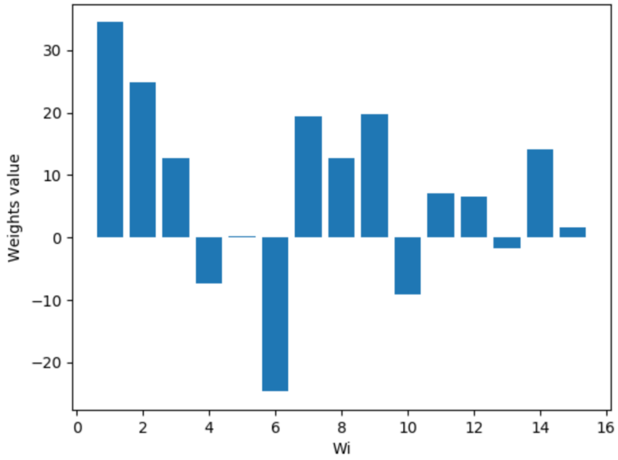
Weights:

#[[34.459106] # [24.83606]# [12.759547]# [-7.2867966]# [0.20280527]

# [-24.666992]# [19.32646]# [12.705511]# [19.707108]# [-9.082143]

# [7.0535207]# [6.526523]# [-1.761117]# [14.1816635]# [1.5776412]]

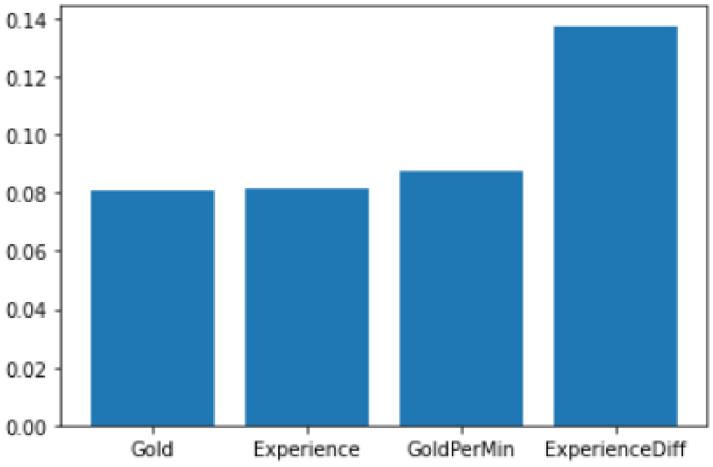
Bias:#[367.81024]



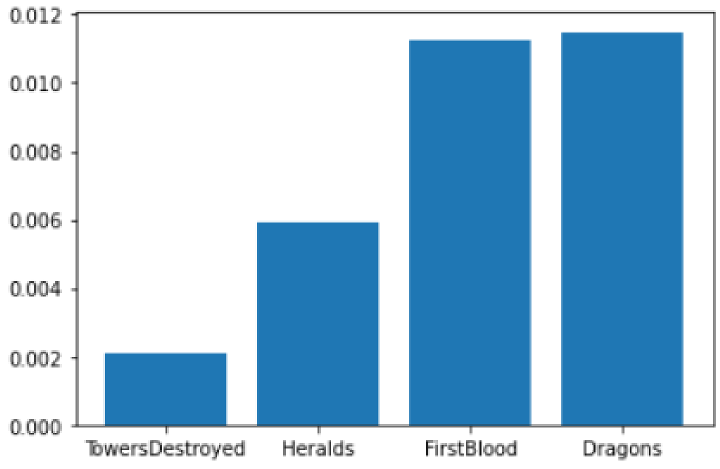
(15 weights)

3.2 Random Forest:

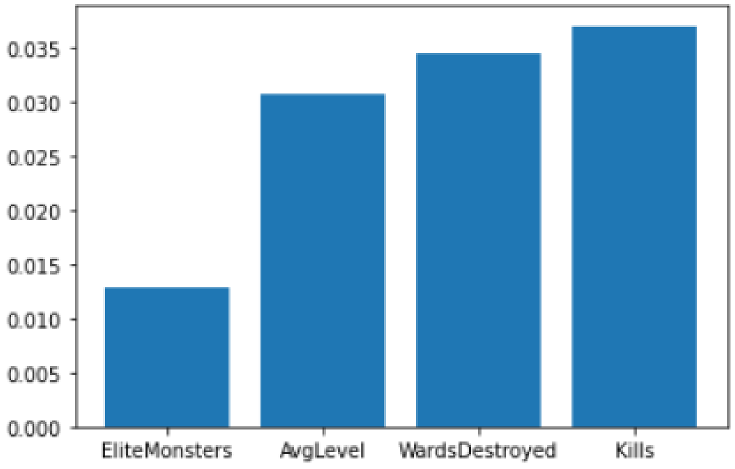
When using Random Forest Classifier to make the model for win prediction based on 10-minute data, we got a model with 72.80% accuracy. But something even more interesting is that the computer gives us a rather special way of viewing the game data. For normal players, killing enemy champions might be their most important focus in the game. People often become excited when their team gets the First Blood(first kill) in the game. But according to our model, First Blood is among the top 4 least important features (Note that we have 20 features in the 10-minute data set.) with a importance of 1.02% (see figure 3.2.2) and even the total kills is not even close to “important”. Total kills in the first 10 minutes has an importance of 3.51%, which sits on the 8th least important feature (see figure 3.2.3). Instead, the data stands out to be important is the total experience difference. Players can get experience from killing minions and enemy champions, but for most times, minions are the most important source of experience. People won’t get the experience if they are not near the minion when it is killed. In other words, the result of the match is surprisingly most influenced by how long and how well people stayed on their lane. Following the experience diff, we can see other stats related to experience and golds, which implies that people are more likely to win the game with good farming instead of fighting with enemies.



Figure(3.2.1). Top 4 most important figures

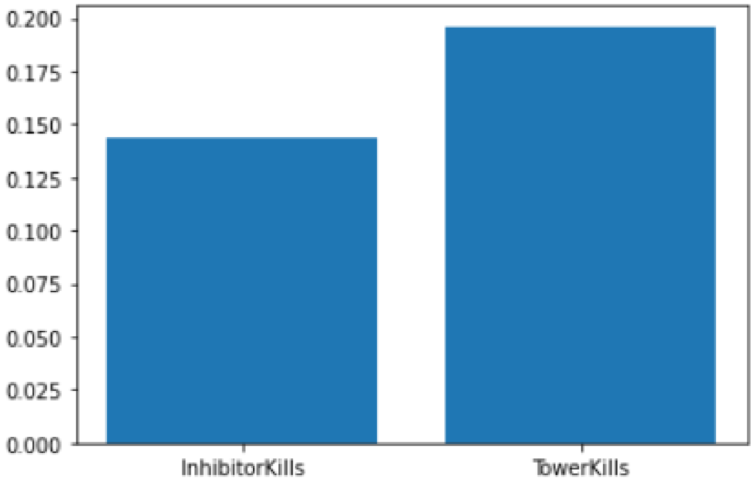


Figure(3.2.2) 4 least important figures

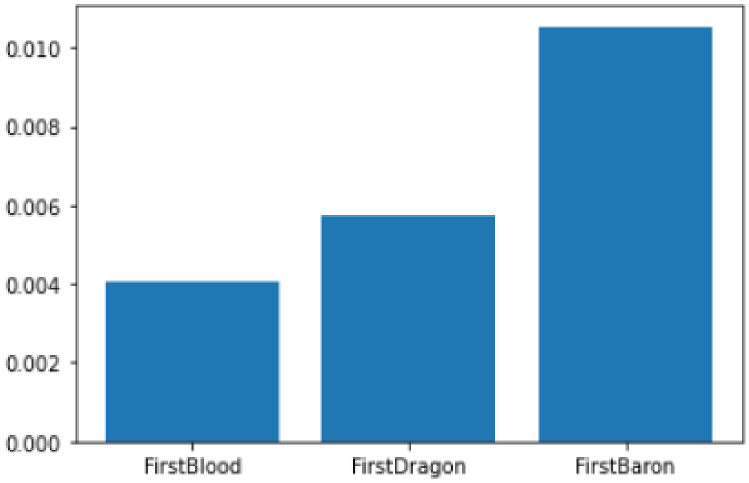


Figure(3.2.3) No.5-8 least important figures

In the process of predicting the game using the end-game data, we found that there's some dominant features and because of these features, the algorithm can easily predict the result. With a basic RFC model, the prediction of the winner has a 97.00% accuracy and with a basic RFR model, the estimation of the gametime has a 97.78% accuracy. What is more interesting than the result is how good the computer recognizes the importance of different data. It's pretty easy for humans to recognize the importance of tower kills because ultimately, the game's purpose is to destroy the nexus and to reach the nexus, you need to destroy the turrets and inhibitors(a small object in front of the nexus). But it's really impressive that the computer can also recognize the fact with the data given to it. Tower kills and inhibitor kills are the two most important features recognized by the RFC model, together having an importance of more than 30% percent (figure 3.2.4). The computer also recognized that the early game objectives are not important to the game result, first blood, first dragon and first baron sit on the 3 least important features with 0.41%, 0.57% and 1.10% importance. First tower is a little more important with an importance of 5.61%. But together, all the early game objectives consist of less than 10% of the final match result.



Figure(3.2.4) Top 2 most important figures



Figure(3.2.4) 3 least important figures

3.3 Finds of HyperParams

When building our model, we found some phenomenon regarding the hyper-parameter of our training model. Unfortunately, because of lack of further research, we did not find a numerical relationship with hyper parameters and the model. However, we found some general theory of how hyper parameters could change the accuracy of a model.

Learning Rate:

We found that when the learning rate is too high, the accuracy of our model increased very fast at the beginning of the learning process. However, when the epoch time increases, the model with a high learning rate cannot fit predicted labels with real labels and the accuracy starts to swing around a specific value.

When the learning rate is two small, the increment of accuracy was too slow that we need more epoches to get ideal results. And the whole process is time-consuming.

Batch Size:

We found that the total training time decreases when batch-size is increased. We also found an increment of GPU ram usage when the batch size is increased.

4. Conclusion & Possible Improvements

4.1 Conclusion

From the works shown above, we can see that the machine learning methods can get us a high prediction accuracy. At the same time, the results shows that the machine learning algorithm can have a deep insight to the game by analyzing the data

4.2 Future improvements

As mentioned earlier, we implemented a series of regression models of machine learning. While considering the project, we found some improvements that could contribute to a more accurate model.

The dataset that we chose was not flexible enough. If we could imply data sets that have “half-game” values and “end-game” values together, it will help build more precise models that predict the end-game data based on in-gaming data, and make the model more valuable.

Moreover, we did not dive deep in some phenomenons we found while completing the project: We simply depicted these phenomenons verbally, instead of implying a formula. We should further develop our project and find potential numerical relationships for those phenomenons.

Citation

Aslan, Mehmet. “League of Legends Worlds 2020 Finals: DWG vs SN”. Nov, 2nd, 2020. https://senpai.gg/blog/league-of-legends-worlds-2020-finals/

Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 0.1038/s41586-020-2649-2.

Hunter, J. D. "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.

Pedregosa, F., et.al. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

The pandas development team. “pandas-dev/pandas: Pandas”. Feb, 2020. Zenodo. https://doi.org/10.5281/zenodo.3509134

Zhang, Aston, et al. “Dive into Deep Learning.” Dive into Deep Learning - Dive into Deep Learning 0.15.0 Documentation, d2l.ai/.