Classifier Performance on League of Legends Matches

Irfan Filipovic
University of Oregon
CIS 472: Machine Learning

Abstract

League of Legends allows teams to forfeit 15 minutes into a match. Players make their decisions based on current game statistics. Using data from a fluctuating game state allows for the outcome to change drastically from the current prediction. The following work utilizes data from ten minutes into a match to predict the outcome using logistic regression and the k-nearest neighbor classifiers. Each classifier is tuned for parameters which will improve accuracies on outcomes. The results show that data drawn from such an early state does not allow for accurate predictions.

1 Introduction

Video games have become prevalent in many demographics as consoles and capabilities are expanded. Many of these games follow a conventional competition between two teams, such as League of Legends created by Riot Games. Following suit of sports teams, there are teams being formed to compete in these games. The University of Oregon currently has five Esports teams in the following games:

1. League of Legends
2. Rocket League
3. Call of Duty
4. Hearthstone
5. Overwatch

Each team has students dedicated to practicing, performing, and winning competitions in their respective games. To track their improvements many look at the statistics provided during and on completion of the match. In League of Legends statistics are tracked by the game and a user may access them at any point to view progress and compare the performance between teams. As a player wins more matches in ranked games, as opposed to casual games, they will climb a ranking system. The ranking system goes from Iron to Challenger, 9 tiers, and each rank has four divisions, 1 being the highest and 4 being the lowest.

Matches take anywhere from 15 minutes to over an hour to complete, noting that at the minimum time a team may choose to forfeit the match following a majority vote. This vote is generally influenced by a player's prediction based on the current statistics. As the game can continue for a much larger amount of time with the tide shifting, it can be hard to determine forfeiting at such an early point. To better evaluate the validity of forfeiting at that time machine learning algorithms can be trained and tested upon existing matches.

The project will utilize logistic regression and k-nearest neighbors classifiers on data from 10 minutes into games of players that are ranked Diamond 1 to Master, ranks 6 and 7. Each classifier will be tuned to utilize parameters that maximize the accuracy, and then the results will be analyzed to determine how valid an option forfeiture is given the statistics. Each match has the possibility of either team making large strides in advantages over the other in short periods of time. Therefore it is predicted that 10 minutes into a match will not be sufficient time to determine a winner.

2 Background

League of Legends main objective to is defeat an enemy structure located at the each teams spawn point.[1] The spawn points at located at opposite corners of a square map. Through the map there are three ”lanes”, one which runs diagonally across the center, and two which follow either perimeter to the other spawn the lanes are connect across the center by a river which a player may walk on. The lanes allow for computer controlled creeps to spawn and march down to fight the other team's creeps, gold is granted to a player that kills a creep. On these lanes each team has 3 turrets and an inhibitor, the inhibitor allows for a team to have a larger minion spawn with each wave of creeps. The turrets will attack creeps and players but may be destroyed by the opposing team for gold.

Gold allows players to purchase items which will increase their effectiveness in the game, whether it be attacking or taking damage from players and creeps. Thus it is important to take towers, creeps, and kill opposing players which also grants gold. Between the lanes there is a jungle which contains monsters, each monster will grant gold and certain ones apply buffs. In the river there is a dragon monster and herald monster which grant gold and buffs improving a teams stance against the other.

A player may view their statistics in game, but will not be privy to all information pertaining to the game. A user may readily see kills, deaths, assists, creep kills, dragon kills, and herald kills when observing stats. These stats correlate to the amount of gold a team has, when a team has more gold to purchase more items they are more likely to win. These are the statistics a player looks at most often to determine if a game will result in a loss, and then decide to forfeit.

Using classifiers to observe this data and determine if an outcome can be predicted on a regular basis will allow us insight into the time-frame of forfeiting at 15 minutes.

[1]Further information can be located at the League of Legends website.

3 Methods

The data contains all possible tracked variables of which some might be difficult to calculate by a player. Therefore the data was split into two sets, one with all variables, and one with easily accessed variables for a player. All variables contains average level and other variables which must be calculated with all player levels, a task difficult for a player in a game.

Implementing the classifiers required me to divide the data into three sets. A test set to train the models on, a development set to test model parameters, and a testing set which allows for accuracy to be determined.

To utilize the data for the classifiers selected it is preprocessed and scaled, so as to avoid large values in logistic regression. The preprocessing is conducted using sklearn's preprocessing which centers the data to its mean for each attribute.

3.1 K-Nearest Neighbors

K-nearest neighbors was chosen as a classifier from the decision tree classifiers due to its quick training.[2] Utilizing the dataset for only the blue team, a trivial choice either team would suffice, the classifier uses 19 attributes under the suggested 20 limit. The classifier will then determine an accuracy using predicted classes.

3.2 Logistic Regression

Logistic regression creates a boundary to determine the class of an example. This will provide insight as to when a team will lose or win, the desired outcome. Using the predicted class we can determine an accuracy for the classifier.

4 Experiments

This section details the methods in which parameters were selected for each classifier, and the resultant values of each parameter. Further evaluation reveals accuracies on the testing data sets. Structure of the implementation of the classifiers followed suit:

1. Convert data into readable arrays.

2. Retrieve desired variables from data.

[2]Learned from CIS 472

3. Build models on all parameter options.
4. Score models on development data set.
5. Select the highest scoring model for each classifier.
6. Analyze accuracy of the classifier on test data.

4.1 Retrieve Data

Data parsed from the data file, see **Reference**, using python library *numpy*. This data was then preprocessed using sklearns preprocessing scale function. The scaled data is split into three seperate data sets as described above. The training set is approximately 70 *percent*, development approximately 10 *percent*, and testing contains about 20 *percent* of the data.

These data sets were then used to train, tune parameters, and analyze accuracy. Once the full data set classifiers finish their computations, data is selected for specific variables and used to implement the smaller classifiers.

4.2 Training Parameters

To train parameters many models were created for each classifier. Each model was then scored using the development data set in order to preserve the integrity using classifiers on the testing set. K-nearest neighbors has a single parameter which was tuned to the development set. The parameter, k, determines how many neighbors are compared while predicting the class. The resulting scores for the full data set are presented in **Table 1**, and the small data set scores display in **Table 2**.

The full data set utilized a k value of 15, and the small data set a k value of 4. Determined by the highest resultant score from the development set.

Logistic regression was tuned for learning rate. The learning rate determines the rate at which weights are updated to reach convergence on the training data set, a lower value results in slow classifying while a high value results in over fitting to the training set. Upon the learning rate the classifier was also tuned for weight regularizers, either L1 or L2. The L1 regularizer takes the sum of absolute weights and is likely to make insignificant attributes have zero weight on the outcome. Allowing for insignificant attributes to impact the outcome slightly

K **K-Neighbors**	**Score**
1	0.65349544
2	0.64336372
3	0.66666666
4	0.68085106
5	0.69098277
6	0.69807497
7	0.68085106
8	0.68389057
9	0.69199594
10	0.69604863
11	0.69300911
12	0.69908814
13	0.70010131
14	0.70010131
15	0.70618034

Table 1: Scores for full data k-nearest neighbors.

K **K-Neighbors**	**Score**
1	0.62208713
2	0.62107396
3	0.66464032
4	0.67983789
5	0.67578520
6	0.67071935
7	0.67375886
8	0.65856129
9	0.67071935
10	0.66970618
11	0.66869300
12	0.66970618
13	0.67578520
14	0.66970618
15	0.67882472

Table 2: Scores for small data k-nearest neighbors.

Reg	LR	Score
L2	1	0.65349544
L2	0.1	0.64336372
L2	0.01	0.66666666
L2	0.001	0.68085106
L2	0.0001	0.69098277
L1	1	0.68085106
L1	0.1	0.68389057
L1	0.01	0.69199594
L1	0.001	0.69604863
L1	0.0001	0.69300911

Table 3: Scores for full data logistic regression.

Reg	LR	Score
L2	1	0.69199594
L2	0.1	0.69300911
L2	0.01	0.68895643
L2	0.001	0.68085106
L2	0.0001	0.67781155
L1	1	0.69199594
L1	0.1	0.69402228
L1	0.01	0.68490374
L1	0.001	0.68591691
L1	0.0001	0.60283687

Table 4: Scores for small data logistic regression.

results from using the L2 regularizer which is the squared root sum of squared weights. This will not make the weights of insignificant attributes reach 0 but approach it.

Resulting scores are located in **Table 3** and **Table 4**. Respectively the full data set model received highest scores with an L1 regularizer and a learning rate of 0.001. The small data set model used the L1 regularizer with a learning rate of 0.1.

4.3 Testing Accuracy

The tuned models are then used to predict outcomes on the testing data set. Predicted outcomes are compared to testing set class values, results are displayed in **Table 5**.

Test	Accuracy
K-nearest Neighbors (FULL)	0.50025316
Logistic Regression (FULL)	0.50177215
K-nearest Neighbors (SMALL)	0.37063291
Logistic Regression (SMALL)	0.51696202

Table 5: Accuracies from each classifier.

5 Conclusion

Our results indicate that we cannot be certain of an evaluation at such an early point in the match. The logistic regression classifier resulted in the highest accuracy on the data set with less variables. While both classifiers had similar accuracies on the full data set, the k-nearest neighbor classifier with less variables was very poor at predicting the outcome.

The outcome for the large data sets coincides with the hypothesis that predicting a winner at such an early state is uncertain. An accuracy of approximately 50 *percent* indicates that half of the matches are predicted incorrectly. Each small data set classifier indicates the same result, such that it is too early to accurately predict a victor. The logistic regression accuracy may be a result of removing insignificant noise variables which do not affect outcome, while regularizers should make these attributes carry smaller weight they may not fully ignore their values. To conduct further studies it may prove fruitful to use classifiers with more variant sets of data from each match, such as using less or more variables. Although the most effective change to improving classifier accuracy may be to increase the time at which data is retrieved from each match, whether that be 25 minutes in or 5 minutes prior to the end of each match. These accuracies can be used to determine the best time at which a victor could be determined.

6 References

Fanboi, Michel's. "League of Legends Diamond Ranked Games (10 Min)." Kaggle, 13 Apr. 2020, www.kaggle.com/bobbyscience/league-of-legends-diamond-ranked-games-10-min.