The Key to Winning in League of Legends

Jeremy Ng Jose Manipon

500882192 5009066166

CPS 844 Datamining

Prof. Cherie Ding

Sunday, April 11, 2021

Brief Description

League of Legends is an online video game with a large competitive scene where various regions around the world compete against each other and have their respective leagues. The dataset for this report uses the competitive matches that occurred between 2015-2017 of all the well-known leagues and competitions including the World Championships. This dataset is very large, it contains nearly all the attributes of the competitive games such as kills, gold, wins, and other various in-game information of over 7,000 matches. The type of data collected from this dataset is numeric and was found searching through Kaggle.

The Problem

League of Legends is a complex game involving various strategies and approaches to win against the enemy team. Winning a game involves many difficult in-game decisions on which objectives to prioritize. This report is done to determine which of those objectives are the most important. These elements include gold, kills, objectives, and other in-game info that will be used as attributes for this report. Using the dataset given, we will determine which of those elements lead to a victory in a game of League of Legends. Answering the frequently asked questions within the community “Do kills win games?” or “Do objectives win games” and so on.

Getting the Data

For our project, we took a League of Legends dataset on all professional matches from 2015 to 2017 from Kaggle. That dataset can be found on Kaggle through this link here: www.kaggle.com/chuckephron/leagueoflegends. We took the data from *LeagueofLegends.csv* and formatted it into an ARFF file using python programming. Each row in the CSV file provides us with two instances in our weka data; one for the winning team and one for the losing team. I selected specific fields for each team that would be relevant to our project. These fields and their conversions can be seen in figure 1 below.

Explaining the Data

Our weka dataset is a numeric dataset that includes 9 attributes: gamelength, gold, kills, inhibitors, towers, dragons, barons, herald, and class.

@ATTRIBUTE gamelength numeric

@ATTRIBUTE gold numeric

@ATTRIBUTE kills numeric

@ATTRIBUTE inhibitors numeric

@ATTRIBUTE towers numeric

@ATTRIBUTE dragons numeric

@ATTRIBUTE barons numeric

@ATTRIBUTE heralds numeric

@ATTRIBUTE class {'yes','no'}

Figure 1. Table showing CSV Fields and Transformation

Field	Field Example	Field Description	Attribute	Attribute Example
gamelength	40	The length of the match in minutes.	gamelength	40
bResult	1	The outcome for the team in a match.	class	yes
goldblue	[2415, 2415, 2711, 3887, 5068, 6171, 7412, 8661, 10154, 11361, 12677, 14558, 15548, 16980, 18324, 19...	The gold that the team has recorded by the minute. For our data, we only need the last element.	gold	62729
bKills	[[10.82, 'C9 Hai', 'TSM Bjergsen', [], 9229, 8469], [16.286, 'C9 LemonNation', 'TSM WildTurtle', ['T...	Every instance of a kill by the blue team represented by [[time, victim, killer, [assists], x, y]]	kills	16
bTowers	[[27.542, 'MID_LANE', 'BASE_TURRET'], [39.269, 'MID_LANE', 'NEXUS_TURRET'], [33.583, 'BOT_LANE', 'IN...	Every instance of a tower destroyed by the blue team in the format [[time, lane, type]]	towers	9
bInhibs	[[36.686, 'MID_LANE'], [29.274, 'MID_LANE']]	Every instance of an inhibitor destroyed by the blue team in the format [[time, lane]]	inhibitors	2
bDragons	[[37.267, None]]	Every instance of a dragon killed by the blue team in format [[time, type]]	dragons	1
bBarons	[[29.255]]	Every instance of a baron kill by the blue team in format [[time]]	barons	1
bHeralds	[[17.102]]	Every instance of a herald kill by the blue team in format [[time]]	heralds	1

Classification Results

1) OneR

```
=== Classifier model (full training set) ===

towers:
	< 6.5	-> no
	>= 6.5	-> yes
(14571/15240 instances correct)


=== Summary ===

Correctly Classified Instances       14571               95.6102 %
Incorrectly Classified Instances       669                4.3898 %
Kappa statistic                          0.9122
Mean absolute error                      0.0439
Root mean squared error                  0.2095
Relative absolute error                  8.7795 %
Root relative squared error             41.9035 %
Total Number of Instances            15240


=== Confusion Matrix ===

    a    b   <-- classified as
 7550   70 |    a = yes
  599 7021 |    b = no
```

The accuracy for our OneR test result was 95.61%. The rules generated for this test were that if the total towers destroyed were greater than or equal to 6.5 turrets then our class value is yes. If the total towers destroyed were less than 6.5 turrets then our class value is no.

2) Naive Bayes

```
=== Summary ===

Correctly Classified Instances       14454               94.8425 %
Incorrectly Classified Instances       786                5.1575 %
Kappa statistic                          0.8969
Mean absolute error                      0.0554
Root mean squared error                  0.2113
Relative absolute error                 11.0814 %
Root relative squared error             42.266  %
Total Number of Instances            15240


=== Confusion Matrix ===

    a    b   <-- classified as
 7412  208 |    a = yes
  578 7042 |    b = no
```

From our results in the Naive Bayes algorithm, the mean of our attribute GAMELENGTH for both class values, yes and no are the same because for every match there is a winner and a loser. For all other attributes, the class value yes has a higher mean. Other notable findings are that the standard deviation of towers for class value no is higher than instances where class value is no because there is an upset potential where a team would be able to destroy all enemy turrets and still lose the game, in the end, resulting in our class value with no in a range of 0-11 for turrets. The same findings apply to gold. Our final accuracy for the Naive Bayes algorithm is 94.84%.

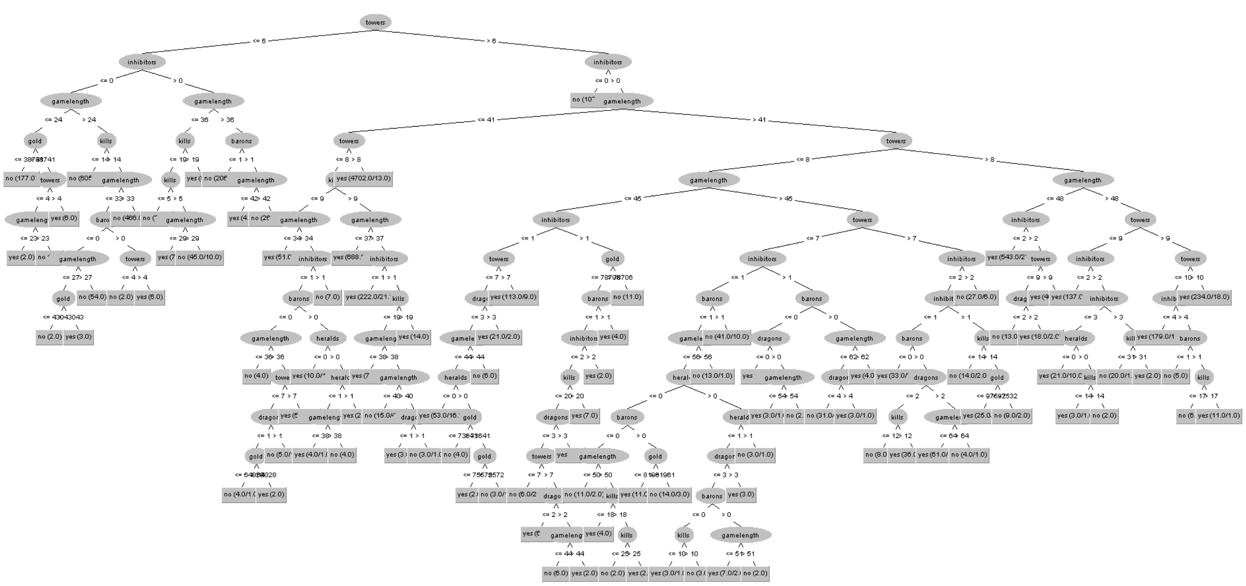
3) J48

a) Pruned:

```
=== Summary ===

Correctly Classified Instances       14765               96.8832 %
Incorrectly Classified Instances       475                3.1168 %
Kappa statistic                          0.9377
Mean absolute error                      0.0436
Root mean squared error                  0.1632
Relative absolute error                  8.726  %
Root relative squared error             32.6433 %
Total Number of Instances            15240


=== Confusion Matrix ===

    a    b   <-- classified as
 7439  181 |    a = yes
  294 7326 |    b = no
```

Pruned accuracy is 96.88%.

Figure 2. Pruned J48 Tree

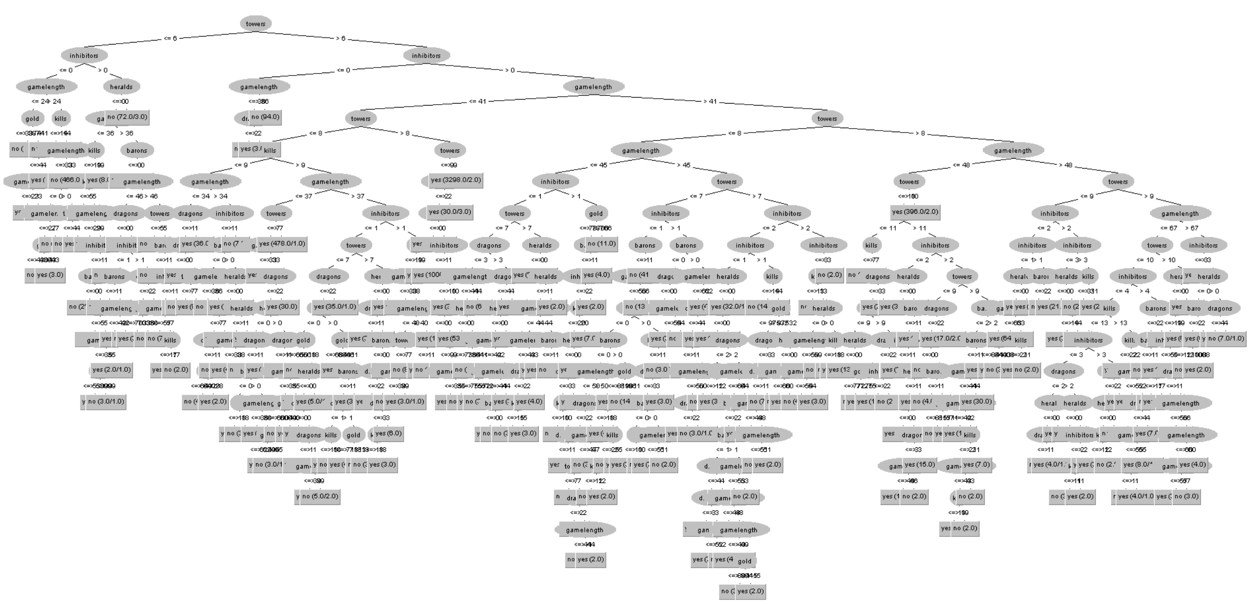


b) Unpruned:

```
=== Summary ===

Correctly Classified Instances       14710               96.5223 %
Incorrectly Classified Instances       530                3.4777 %
Kappa statistic                          0.9304
Mean absolute error                      0.0405
Root mean squared error                  0.1727
Relative absolute error                  8.1074 %
Root relative squared error             34.5352 %
Total Number of Instances            15240

=== Confusion Matrix ===

    a    b   <-- classified as
 7396  224 |    a = yes
  306 7314 |    b = no
```

Unpruned accuracy is 96.52%.

Figure 3. Unpruned J48 Tree



4) Logistics

```
=== Summary ===

Correctly Classified Instances       14722               96.601  %
Incorrectly Classified Instances       518                3.399  %
Kappa statistic                          0.932
Mean absolute error                      0.0505
Root mean squared error                  0.1597
Relative absolute error                 10.0942 %
Root relative squared error             31.9393 %
Total Number of Instances            15240

=== Confusion Matrix ===

    a    b   <-- classified as
 7401  219 |    a = yes
  299 7321 |    b = no
```

The accuracy for our Logistics test result was 96.60%.

5) Multilayer Perceptron

```
=== Summary ===

Correctly Classified Instances       14827               97.29   %
Incorrectly Classified Instances       413                2.71   %
Kappa statistic                          0.9458
Mean absolute error                      0.0394
Root mean squared error                  0.1429
Relative absolute error                  7.8736 %
Root relative squared error             28.5797 %
Total Number of Instances            15240

=== Confusion Matrix ===

    a    b   <-- classified as
 7423  197 |    a = yes
  216 7404 |    b = no
```

The accuracy for our Multilayer Perceptron test result was 97.01%.

Figure 4. Table showing final accuracy for the above tests.

Classifier	**Accuracy**
OneR	95.61%
Naives Bayes	94.84%.
J48 Pruned	96.88%
J48 Unpruned	96.52%
Logistics	96.60%.
Multilayer Perceptron	97.01%.

Our most accurate test is the Multilayer Perceptron at 97.01% accuracy.

Selected Attributes Results

1) InfoGain Algorithm + Ranker Search Method

The test results received are as follows:

```
Ranked attributes:
 0.8205   5 towers
 0.755    4 inhibitors
 0.4096   3 kills
 0.3787   7 barons
 0.2693   6 dragons
 0.141    2 gold
 0.0196   8 heralds
 0        1 gamelength
```

The results we received running the selected attribute algorithm, InfoGain, shown above state that the towers are the most important objective in the game. It is ranked first out of all the other listed attributes in the test. This makes sense, to win a game you would need to destroy a minimum of 5 turrets and that's the best-case scenario. It also ranks inhibitors as the second most important objective because like turrets to win, a team would need to destroy a minimum of 1 inhibitor. These two objectives are required to be destroyed and are mandatory to win, unlike the other objectives.

2) Correlation Algorithm + Ranker Search Method

```
Ranked attributes:
 0.881   5 towers
 0.708   4 inhibitors
 0.64    3 kills
 0.595   7 barons
 0.551   6 dragons
 0.329   2 gold
 0.161   8 heralds
 0       1 gamelength
```

Running the correlation algorithm provides us with similar results as the ones we received using InfoGain. The numbers provided differ, where the listed attributes have increased in value. The attributes ranked higher, received a larger value and are given more importance compared to the results generated using info gain.

Conclusion

In conclusion, we formatted a CSV dataset from Kaggle into an ARFF file to be used in Weka. Using python programming we were able to transform the CSV data into useful data for this assignment. We tested various algorithms and found that the multilayer perceptron is the highest accuracy test with 97.01% correctness. This confirms our classification using wins and losses as our class attribute. We also ran two selected attribute algorithms which both gave the same ranking results. Finally, the top three attributes are towers, inhibitors and kills.