Machine Learning Final Report

1. **Introduction**

Competitive videogames are a collection of tens, hundreds, or even thousands of factors, all contributing to the outcome of the game as a whole. But how can we know which of these factors are most important, or which aspects should be prioritized over one another? By taking a machine learning approach, we can analyze these factors in attempt to see how they contribute to a game’s outcome, and to ultimately provide knowledge on how to be better at the game.

1. **Problem Definition and Algorithm**

2.1 Task Definition

Inputs: Factors of gameplay from different competitive games (Call of Duty and League of Legends)

Outputs: a prediction of ‘Win’ or ‘Lose’, weighing the factors of gameplay post-mortem to try and determine whether a team won or lost. We can then see what information the machine learning algorithm prioritized in order to gain insight into which factors of the game are truly important, bettering ourselves as players.

2.2 Algorithm Definition

Our algorithm can be broken up into three steps: processing the input data, training various classifiers, and analyzing the accuracy and other important qualities of those classifiers. All of the data that served as input needed to be processed before it could be used by the classifiers. As such, the datasets were converted into lists of workable numerical data and broken down to be more manageable. Each data entry from the League of Legends data was put into two different lists, one containing feature data focused on individual contribution and another containing feature data focused on team contribution. For the Call of Duty data, the entries did not have enough data regarding team contribution to be split as the League of Legends data was. Instead, Feature Elimination was used to reduce the number of features being considered to those that have a stronger relationship with the class label. The datasets composed of the remaining features were then used to train various classifiers. Algorithms used were Decision Trees, Random Forests, Boosted Trees, Naïve Bayes, SVM, and Perceptron based Neural Network. The majority of these were chosen because they can be analyzed. For instance, you could look at what splits a decision tree chooses to make to see which factors were most important. Similarly, you can analyze the weights of a Naïve Bayes algorithm to see which factors weigh the most heavily on the classifier’s decision

1. **Experimental Evaluation**

3.1 Methodology

3.1.1) *What criteria did was used for evaluation?*

We split data into train and test, and then calculated the accuracy of our predictor on the test data. We also placed an emphasis on understandability, as one of the goals of this experiment was to achieve some new understanding of the game. Hence, we tended to use more inspectable algorithms rather than black boxes.

3.1.2) *What specific hypothesis did this experiment test?*

We went into this experiment looking at two games: Call of Duty and League of Legends. One data set we had was rather simple, involved the number of Kills, deaths, assists, and a few other aspects, mostly individual. The second data set was richer, involving many aspects of individual as well as higher macro-level team play. We suspected that this type of data would give us better results, and with a more complex game we could gain a more complex insight. In addition to this, we hypothesized that splitting the second game into two datasets (individual based, team based) would result in one type of gameplay (playing for the team vs playing for yourself) being better overall.

3.1.3) *What are the dependent and independent variables?*

The probability of victory (or rather, the classifiers probability of correctly guessing) is dependent on the other aspects of gameplay (Kills, Deaths, Assists, Amount of gold collected, Number of objectives captured, amount of damage dealt, etc...)

3.1.4) *What training and test data was used?*

We used real game data to train and test on, played by real people. Hence this could not be more realistic, and it is interesting as it gives us the ability to guess how the game would play out differently had small aspects changed. The League of Legends datasets are from non-professional ranked matches while the Call of Duty datasets are from professional tournaments. These datasets were divided up into training and testing datasets, with the majority of the data being used for training. Which entries are put into which group is determined randomly.

(see also matches1.json – matches10.json, data-2018-01-14-neworleans.csv, and data-2018-04-01-birmingham.csv) (.json files can be viewed through online json parsers)

3.1.5) *What performance data was collected, and how is it being presented?*

Data was collected in the form of overall accuracy, as well as factor importance for each algorithm. In addition to this, we have a visual representation of our decision tree algorithm’s different results as well.

3.2) Results

3.2.1) Results for Call of Duty Data

Call of Duty was the less detailed of the two data sets. Nevertheless, it was still quite descriptive and we were able to reach an accuracy between 60 and 70 percent.

|  |  |
| --- | --- |
| Algorithm | Sample Accuracy |
| Naïve Bayes | 0.6885 |
| Support Vector Machine | 0.7621 |
| Neural Network | 0.6837 |
| Decision Trees | 0.6378 |
| Random Forests | 0.6891 |
| Boosted Forests | 0.6858 |

Some of the algorithms we used also allowed us to gain information about which features had the greatest impact on the results. For instance, based on data from the Boosted Tree classifier, k/d ratio and hill time (s) were two of the most important features in determining victory in the Hardpoint game mode of Call of Duty.

(see also, cod\_tree.pdf and output.txt)

3.2.2) Result of League of Legends Individual Data

League of Legends Individual Data was more detailed, providing us more tools to properly train our classifier. Using individual data, we were able to achieve between 70 and 80 percent.

|  |  |
| --- | --- |
| Algorithm | Sample Accuracy |
| Naïve Bayes | 0.7126 |
| Neural Network | 0.786 |
| Decision Trees | 0.7775 |
| Random Forests | 0.809 |
| Boosted Forests | 0.8265 |

Some of the algorithms we used also allowed us to gain information about which features had the greatest impact on the results. For instance, based on data from the Random Forest classifier, damage dealt to Objectives was easily the most important feature in determining victory.

(see also, league-team.pdf, League\_individual.png, and output.txt)

3.2.3) Result of League of Legends Team Data

This data ended up being our most accurately classified, clocking in at 90.0%. This goes to show that League is inherently a team game, and thriving often involves being a proper part of the team

|  |  |
| --- | --- |
| Algorithm | Sample Accuracy |
| Naïve Bayes | 0.896 |
| Support Vector Machine | 0.868 |
| Neural Network | 0.898 |
| Decision Trees | 0.875 |
| Random Forests | 0.9 |
| Boosted Forests | 0.875 |

Some of the algorithms we used also allowed us to gain information about which features had the greatest impact on the results. For instance, based on data from the SVM classifier, firstInhibitor (or whether or not this team was the first to destroy an enemy’s inhibitor) is one of the more important features for determining victory.

3.3) Discussion

3.3.1) *Was our hypothesis supported?*

Yes. League data proved to be better for classifying, and there was a difference between the accuracy of team based and individual aspects

3.3.2) *How can the results be explained in terms of the underlying properties of the algorithm and/or the data?*

Call of duty is inherently more of a mechanical game (although some positioning and other skills hard to quantify in a data set do affect this), and hence it is harder to properly evaluate based on individual performance. League on the other hand, is much more focused on macro, in a way this is easily quantifiable in a data set (objectives captured!). This is likely why League’s classifier had a better time than that of Call of Duty. In addition to this, League’s *team* classifier did better than leagues individual classifier. This is because these team plays and objectives are important, and as previously stated, League of Legends is inherently a very team-based game. Also, the features that could be interpreted as being the most important for the various classifiers make sense. For Call of Duty, k/d is a general indicator of performance in a match for any game mode, while features like hill time (s) are directly related to the objective the players must pursue to win a Hardpoint match. As for League of Legends, damaging objectives is required to win the game and destroying inhibitors often give a team the momentum they need to win. Lastly, one very interesting observation made regarding League’s Individual-based decision tree:

It seems as if the decision tree learned about support roles, without being explicitly taught. There exists a path on the decision tree that analyzes Deaths, then Gold Earned, then Assists, and then Damage Mitigation. Typically, one would think that many deaths, and little gold earned would be a bad thing. However, there exists a role in League of legends called the support. The job of the support is to babysit fighters that start off weak but scale harder into the late game. A support should funnel gold into the fighter they are protecting instead of collecting it themselves and should often be prepared to die to protect their partner. This is mirrored in the decision tree, as the path of (low kills, low gold earned, high assists, high damage mitigation) is mostly positive, as this mirrors the function of a good support. Nowhere was this explicitly taught to the classifier, and to the untrained player, it seems almost unintuitive.

1. **Related Work**

1) Many professional League of Legends teams do video reviews of professional matches, in order to gain a better understanding of what could have been done better each game. This differs fundamentally from the means of analysis we used (human vs machine)

2) A select few players choose to cheat while playing Call of Duty and League of Legends by employing bots, but this is different as it focuses very heavily on mechanical gameplay, basically playing to a level that is impossible to achieve by a human, effectively being able to ignore macro altogether by outclassing opponents so hard mechanically. This is not feasible for a human to achieve, as at a high enough level of gameplay macro will always hold a higher importance

1. **Future Work**

*1) What are the shortcomings of the current method?*

The call of duty model is restricted by the limitations of the data set – there are larger macro-level elements that go into Call of Duty (albeit less than League of Legends), but those are much harder to quantify, as they involve positioning and communication, and this is hard to put into a table. The positioning could be improved upon by using a heatmap or something of the like

**6. Conclusion**

In this experiment we showed a few things. Firstly, how the quality of data expression can affect the quality of one’s machine learning algorithm. Second, we learned that team-based aspects of League of Legends were more important, winning out over individual level play. Lastly, we showed that our ML algorithm was capable of learning oddly specific pieces of gameplay that almost seem unintuitive to the untrained eye. This was exciting to see and was a testament to how great many of these algorithms can be.