Jack R. O’Connor

IST 652

Final Project Report

14 September 2020

Final Project Report

League of Legends is an online computer game with a massive player-base. Each game consists of two teams of five players trying to reach the other team’s base and destroy their “nexus”. The teams must play defense and offense simultaneously to achieve the goal: keep the enemy out of their base while infiltrating and destroying the other base. The beginning of each game is always the same in which each player chooses a character to control and is then thrown onto the battlefield at level one with no items. Players must gain experience and gold in order to level their character up and purchase items that give their character more stats like damage and health. Players gain gold and experience by killing non-playable characters called minions or by killing enemy players.

League of Legends is a treasure trove for data enthusiasts even if one is not familiar with the game itself. Because League is a complex game with over 145 playable characters (each with 4 unique abilities), over 175 unique items to purchase in-game, and ten human players on the battlefield at a time, the analysis of game data can be as simple or as complex as the researcher would like; however, no matter how complex and accurate a model becomes, it is extremely difficult to incorporate the human decision-making into a quantitative model.

This project uses data that is a snapshot of game statistics from the ten-minute mark of 10,000 ranked games played at the Diamond I to Master rank (top 0.25% of players). The average game of League takes about 30 minutes, so looking at data from such an early point in the game allows for many events to occur in the second two-thirds of the game to go unaccounted for. Generally speaking, the team that gets ahead of the other team early on in the game has a great advantage both level- and item-wise; however, the tide of the game can always shift at any point due to good or bad decision making. Therefore the data shows shifts in “control” over the game meaning one team is ahead at the ten-minute mark but does not end up winning. In this sense, the data is limited by what happens in the time that happens from the ten-minute point through the end of the game.

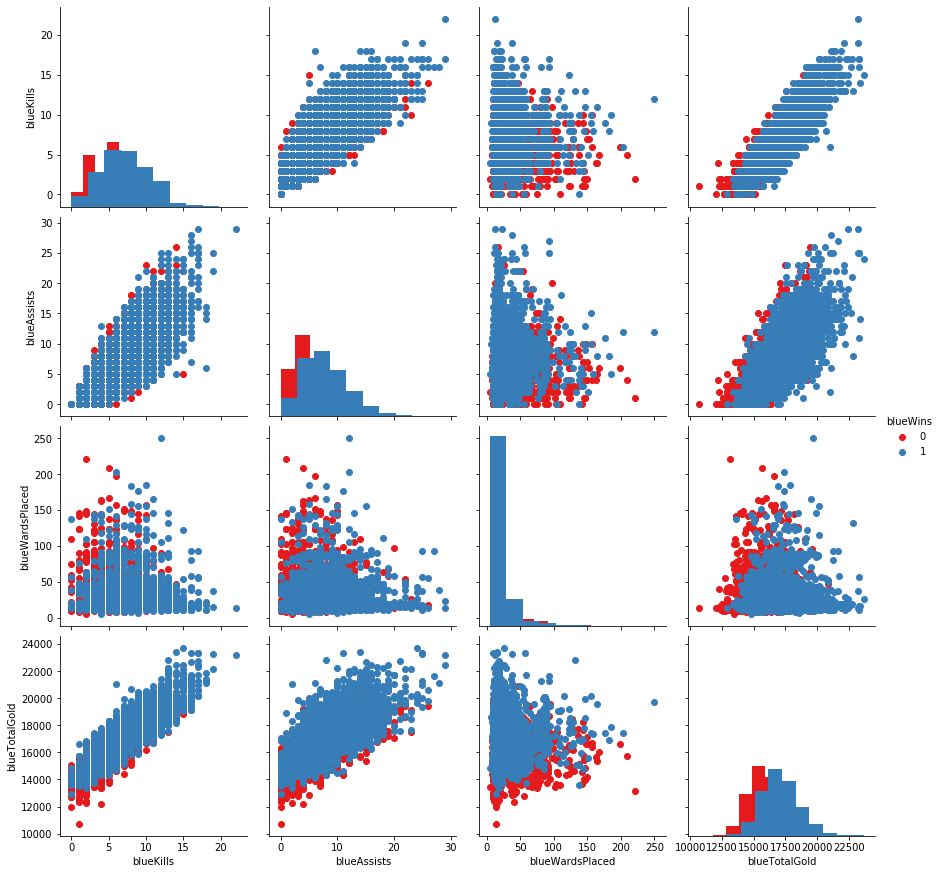
Taken from Kaggle, the League data contains 19 variables for each team (blue and red) which are listed in the below table with a brief description of each one.[[1]](#footnote-1) Only the blue variables are listed, but note that they are mirrored for the red side.

|  |  |
| --- | --- |
| gameID | Unique identifier for each game given by the API |
| blueWins | The classifier variable. 0 for a blue loss and 1 for a blue win |
| blueWardsPlaced | How many vision-granting wards the blue team placed |
| blueWardsDestroyed | How many enemy vision-granting wards the blue team destroyed |
| blueFirstBlood | 0 if blue team did not get the first player kill and 1 if blue did get the first player kill |
| blueKills | How many kills the blue team had at 10 minutes |
| blueDeaths | How many deaths the blue team had at 10 minutes |
| blueAssists | How many assists the blue team had at 10 minutes |
| blueEliteMonsters | How many elite monsters the blue team had killed (Dragons + Heralds) |
| blueDragons | How many dragons the blue team has slain (maximum of 1) |
| blueHeralds | How many heralds the blue team has captured (maximum of 1) |
| blueTowersDestroyed | How many enemy towers the blue team has destroyed |
| blueTotalGold | The total amount of gold the blue team has acquired |
| blueAvgLevel | The average level of all five players on the blue team |
| blueTotalExperience | The total experience gained of all five players on the blue team |
| blueTotalMinionsKilled | The number of minions killed by the blue team |
| blueTotalJungleMinionsKilled | The number of jungle minions killed by the blue team |
| blueGoldDiff | The difference in gold between the blue team and the red team |
| blueExperienceDiff | The difference in experience between the blue team and the red team |
| blueCSPerMin | The number of minions killed by the blue team per minute (blueTotalMinonsKilled divided by 10) |
| blueGoldPerMin | The amount of gold earned per minute by the blue team (blueTotalGold divided by 10) |

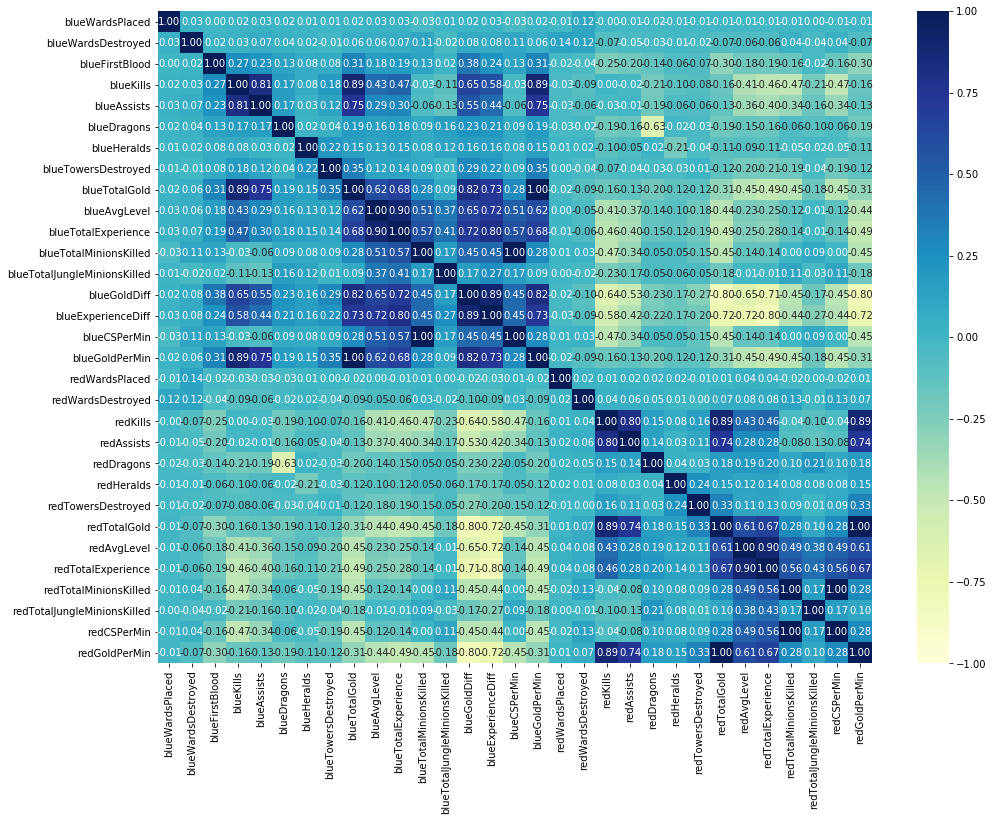
The goal of this project is to predict whether the blue team will win a game based on the data taken at the ten-minute mark of a game, but before any models can be trained, this data needs to be explored thoroughly and cleaned up.

The first step in the cleaning of the data was to drop some redundant columns. What is meant by this is any variable that was wrapped up in another variable like EliteMonsters. Since this was just an aggregate column of Dragons plus Heralds it was removed completely. Further, the deaths columns were removed because they were represented in the kill columns for the opposite teams (i.e., red deaths equal blue kills and vice versa). The following columns were dropped: gameId, redFirstBlood, blueDeaths, redDeaths, redGoldDiff, redExperienceDiff, blueEliteMonsters, redEliteMonsters.

To begin the exploration, the following visualization was created to show each “blue” variable plotted against the other variables. The colors represent the victors of each game (blue indicates a blue victory and red indicates a red victory).



Even though this data set is clean in the sense that there are no NULL values or any outliers to account for, there is an issue of multicollinearity within this data. Even without prior knowledge of the game, it is easy enough for one to see that several of the variables are “rolled-up” into another aggregate variable or broken down into a different metric based on time. For example, blueEliteMosters is just the blueDragons plus the blueHeralds. Similarly, the blueCSPerMin is just the blueTotalMinionsKilled divided by ten minutes. The multicollinearity is apparent in the above visualization as several of the graphs have linear upward trends. To quantify these relationships, a correlation matrix was created which was used to start cleaning up the data to eliminate the multicollinearity.



The correlation matrix above shows just how correlated the independent variables are in this data set. Focusing on the dark blue or light-yellow cells, it is easy to get a sense of just how related each variable is to another. With prior knowledge of the game, this correlation matrix makes perfect sense. Kills and Assists are highly correlated with each other and both are correlated to total gold. This is because a lot of the time several of the players work together to get a kill on the enemy. Only one person is awarded the kill, but the others are awarded an assist. Kills and assists both earn players gold.

A few of the variables were dropped all together while a few were combined. The columns that were dropped are blueAvgLevel, redAvgLevel, blueTotalGold, redTotalGold, blueTotalExperience, redTotalExperience, redCSPerMin, blueCSPerMin, redGoldPerMin, and blueGoldPerMin. The theory behind dropping total gold and total experience was that these were probably the two most important independent variables; however, they did not give any indication of how a team can win a game. To further explain, it makes intuitive sense that most of the time the team with the most experience and gold at the ten-minute mark because this is the team that is doing better at that point in the game. And since kills, assists, and minion kills all give gold and experience, the gold and experience values are represented in the other variables. By removing the total gold and experience for each team, the models will be able to rank feature importance and give players real insight into what wins games. To illustrate the dominance of total gold on the models, please see the following chart created to show feature importance of a random forest model trained with total gold left in.



To deal with the correlation between the assists and the kills, each assist was assigned a value of 0.5 kills and added to the respective kill column. This decision was made because the gold given for each assist is 150 and a kill alone is worth 300 gold.[[2]](#footnote-2) There is a “shutdown” mechanic in the game that gives more gold for a kill on a player that is further ahead, but since that granularity is not represented in this data set, an assist counting as half of a kill is sufficient. After the assists were added to the kills columns, the assists columns were dropped.

Once the data was cleaned up and the multicollinearity resolved, the variables most correlated to the classifier (blueWins) were pulled to be features in the models. Any column that had a correlation value less than -0.2 or greater than 0.2. The resulting list consisted of blueFirstBlood, blueKills, blueDragons, blueTotalMinionsKilled, redKills, redDragons, redTotalMinionsKilled. Once these columns were chosen, the data was divided into a training set and a testing set. The test set size was set to .20 and the random\_state was designated at 42 for reproducibility. All of the data was then normalized using the MinMaxScaler() function from the sklearn package.

The first model trained was a random forest model through the Scikit learn (sklearn) package.[[3]](#footnote-3) A grid of parameters was created with five “n\_estimators” and 3 “max\_depths” for the random forest to run through. The GridSearchCV finds the most efficient set of parameters in the grid and trains the model using them. After training was complete, the model was applied to the test set and achieved a 72.98% accuracy. The feature importance was then plotted and is displayed in the bar chart below.



It is worth noting that the Gold variable had a much more significant role in the model than the Experience feature did. This means that going for plays that net higher values of gold are usually more optimal to winning the game than plays that net higher values of experience. Of course, this is a generalization and every game has different situations, but it seems as though gold is more important than experience.

A second random forest model was trained in the same way without the blueExperienceDiff and blueGoldDiff variables. The purpose of this model was to see more granularity to the feature importance. This model sought to answer the question of what game actions contribute the most to the outcome? This model sacrificed some accuracy and ended up with a 71.26% accuracy when applied to the test set; however, it showed two different important features than the previous model.



Kills and minion kills are the most important things pre-ten-minutes in the game. Dragons, towers, and first blood do not really contribute much to the model. This coincides with the previous findings that showed that the gold difference was the most important feature. Kills and minion kills are the fastest way to get gold and this chart shows that minion kills are slightly more important than player kills. So, according to the random forest models, players need to focus on killing minions in the early game to have a major impact on the outcome of the game.

The next model that was built was a Naïve Bayes classification model. This model achieved a 73.68% accuracy which is the most accurate of the three, so far. There is not much else to note regarding the Naïve Bayes model as I was not able to extract the coefficients for each feature from the model.

The final model was a logistic regression that was able to achieve a 72.87% accuracy on the test set. From this model, the coefficients of the features were extracted and put into a dictionary (note that this model does contain the blueGoldDiff and blueExperienceDiff variables). The following table shows the coefficients:

|  |  |
| --- | --- |
| blueGoldDiff | 6.176955161632731 |
| blueExperienceDiff | 4.856234745702455 |
| blueKills | 0.7572257395889194 |
| blueDragons | 0.31498255784923257 |
| blueFirstBlood | 0.1338586762690842 |
| redTotalMinionsKilled | 0.06089973880909091 |
| blueTotalMinionsKilled | 0.00816553853218096 |
| redDragons | -0.22166517792746954 |
| redKills | -0.8108904811366964 |

This list is extremely interesting in the sense that it places minion kills as a minor contributor to the model. Gold is still king in this model by far, but the logistic regression places much more emphasis on dragons than the random forest models. Excluding the blueGoldDiff and blueExperienceDiff columns, blueKills, redKills, and blueDragons are the top three features in this model, while the minion kills columns are the bottom two.

Overall, the findings of this analysis proved as complex as the game itself. One model shows that minion kills are integral to the success of the team while another model barely even takes them into account. One constant is that Gold is always an important feature no matter the model. In order to win the game, players need to collect more gold than their enemies and use that gold to “snowball” their advantage.

Because the most accurate model only reached 73.68% accuracy, one conclusion of this analysis can be that there is a significant portion of games whose outcomes are not determined by the ten-minute mark. This is where the human decision-making comes into play. How does one model poor choices or mechanical miss-plays? How does one measure emotions or how high tensions get during a game? All of these are impossible with the current data set, but surely there is a fan out there that will take League of Legends stats to MLB level.

1. <https://www.kaggle.com/bobbyscience/league-of-legends-diamond-ranked-games-10-min> [↑](#footnote-ref-1)
2. <https://tinyurl.com/y6op2rp3> [↑](#footnote-ref-2)
3. <https://scikit-learn.org/stable/> [↑](#footnote-ref-3)