LOL Diamond Ranked Games Outcome Prediction

Xin Chen, Ruiming Yu

1 INTRODUCTION

The aim of this project is to predict the outcome of matches in the Diamond Ranking game of League of Legends using machine learning methods. League of Legends is a multiplayer online battle arena video game where two teams, blue against red, of five champions face off to destroy the enemy's Nexus located in the middle of their base. Champions are manipulated by players, and they can build their powers by stronger items and levels from gold and experiences by destroying turrets, killing minions, jungle minions, and enemy champions.

Other natural resources such as Rift Heralds and dragons can also affect the game. It usually takes about 20 to 40 minutes for a single game. The beginning of the game could bring a great impact on the result due to the snowball effect, so for players who want to rank up can pay more attention to these key features. The winning team and which in-game actions in the first ten minutes are the most relevant and decisive to winning the game are predicted using data from the first 10 minutes. Since the response is a binary variable, classification models including Logistic Regression, LDA, QDA, KNN, Random Forest, and Decision Trees will be used in this project. After evaluating all the models, we concluded that logistic regression was the best model, and gold difference, dragon-slaying and experience difference were the three most important features.

2 RELATED WORK

League of Legends is a typical game which is easy to get started but hard to master. Players with higher rankings focus on some details that are treated not important for those with low rankings. As a result, it is hard to define the key strategy to winning by interviewing high ranking players, as they be may unaware of some of the hidden game-changing features. However, for a player who wishes to improve his skills, the only seemingly possible way is to listen to few people, which could be biased.

Therefore, we try to build machine learning models to find out what really matters in determining the winning side of a game by investigating the correlations in these features from the first 10 minutes of game. Instead of searching for those so-called winning tricks from only one or a few players, we plan to summarize some key data-driven factors from almost 10,000 high-ranking games.

3 METHODS

There are 21 predictors used in this project, they are blue kills, blue wards placed, blue wards destroyed, blue dragon, blue tower destroyed, blue total jungle minions killed, blue experience difference, blue first blood, blue assists, blue heralds, blue average level, blue gold difference, red kills, red wards placed, red wards destroyed, red assists, red dragons, red heralds, red tower destroyed, red total jungle minions killed, and red average level. All the predictors are continuous variables. The response of this project is blue wins and it is a categorical variable; 1 is used for victory and 0 is used for defeat. There were six machine learning models used in this project, they were Logistic Regression, LDA, QDA, KNN, Random Forest, and Decision Trees.

PCA was conducted first to summarize the set with a reduced number of representative variables that collectively explain most of the variability in the original dataset. However, the result shows that there was no significant drop in the proportion of variance, so PCA was not an ideal method for this project. Since the response is a binary variable, Logistic regression is a straightforward method we can think of. It is evaluated by the outcome greater than 0.5 or not. Any outcome larger than 0.5 would be considered a victory for the blue team and less than 0.5 would be considered a defeat for the blue team.

We trained six models using different supervised machine learning methods for this classification problem. For each model, we applied the cross-validation methods by training the model with 4 of the 5 folds in the training set, and computed the cross-validation error on the last fold of the training set. Generally we used the 'caret::train()' function to proceed the cross-validation step. We repeated this methods till we have an optimal parameter in the model. For complex models such as Random Forest, we only tuned several most important parameters such as ntree and mtry. In the last step, we tested the well-trained model on the test set, and recorded the training error, test error, ROC curve and AUC score for model comparison to select the best model.

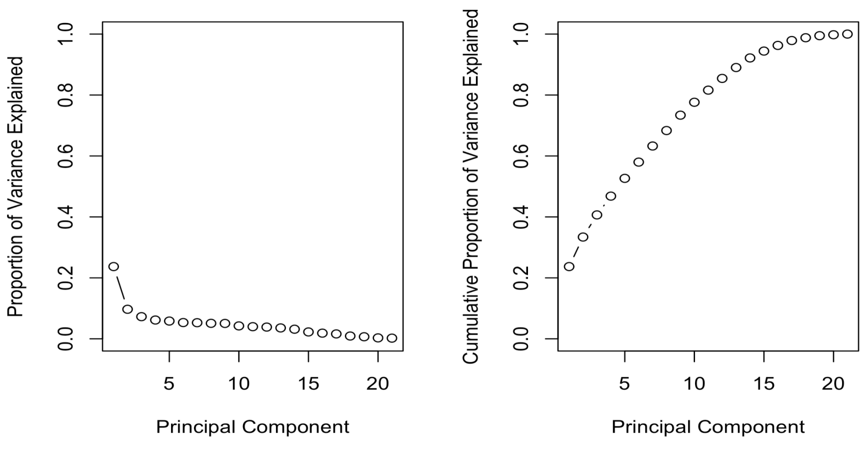

Figure 1: PCA result

4 DATA and EXPERIMENT SETUP

The original dataset contained 9,879 observations and 40 columns. Except for the response, buleWins, which is a binary variable, other variables are all continuous variables and worked as predictors in the models. No missing data was detected in the dataset. maximum and minimum values were also checked for abnormal values and no abnormal value was found in the dataset. A heatmap was plotted to check the correlation between variables. Collinearity was detected in the dataset when some of the columns were completely negatively related such as redGoldDiff and blueGoldDiff or redKills and blueDeaths. The variable related to blue team would be left in each pair for such pairs in order to prevent collinearity.

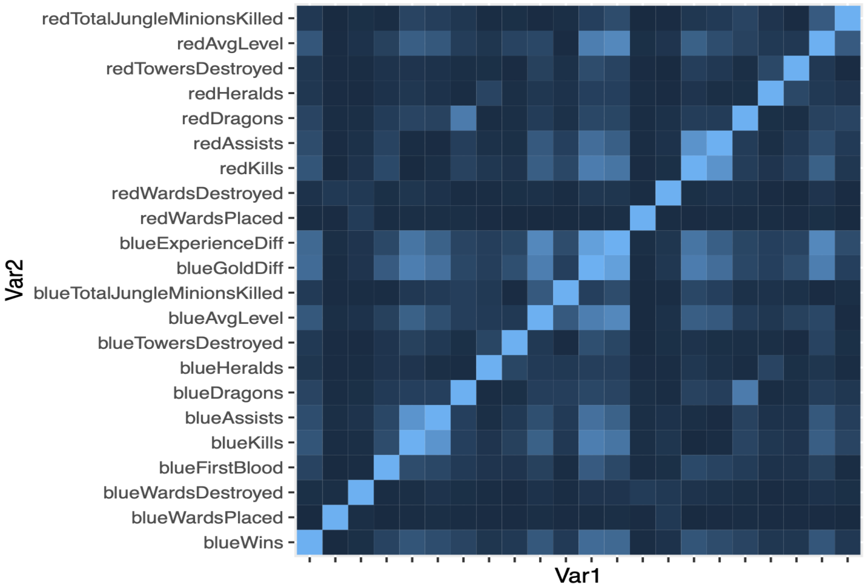

Figure 2: Heatmap plot

Since gameId was irrelevant to the research, we also removed this column. The cleaned dataset contained 9,879 observations and 22 columns. Within the 22 columns, one variable was the response and 21 variables are the predictors. After the data cleaning part, the dataset was split into training and test sets followed by a ratio of 8 to 2. The training set was then split into k-fold cross-validation set using a k = 5 to evaluate the performance of models.

It is worth noting that many players would try to predict the winner by only the differences in kills and they believed the side with more kills was more likely to win the game. Therefore, this idea was applied to our naive model as the baseline evaluation. The blueWin was the response in the model, and blueKills and redKills were selected as predictors in the naive model. The training and test errors of the naive model were 0.2967 and 0.2900. Both errors were around 0.3, which made this model reasonable but not an excellent predictor, so different models were conducted to improve the prediction efficiency as follows.

5 RESULTS

After running all the models, we got the training errors, test errors, and AUC scores. The results are presented in the following table (Table 1).

Model	Training error	Test error	AUC score
naive	0.2967	0.2900	0.7097
logistic	0.2667	0.2733	0.8126
LDA	0.2667	0.2773	0.8124
QDA	0.2884	0.2869	0.7750
KNN	0.2684	0.2869	0.7930
Decision tree	0.2742	0.2763	0.7238
Random forest	0.2033	0.2788	0.8005

Table 1: Training error, test error and AUC score of six models

All the training error, test error, ROC, and AUC from all six models with the naive model and found that all six models are all improved compared with the naive model. By comparing the training errors and test errors, the training accuracy score of Random Forest was the best and the test accuracy score of Logistic Regression was the best indicating the test error of Logistic regression was the smallest in figure 3 and figure 4.

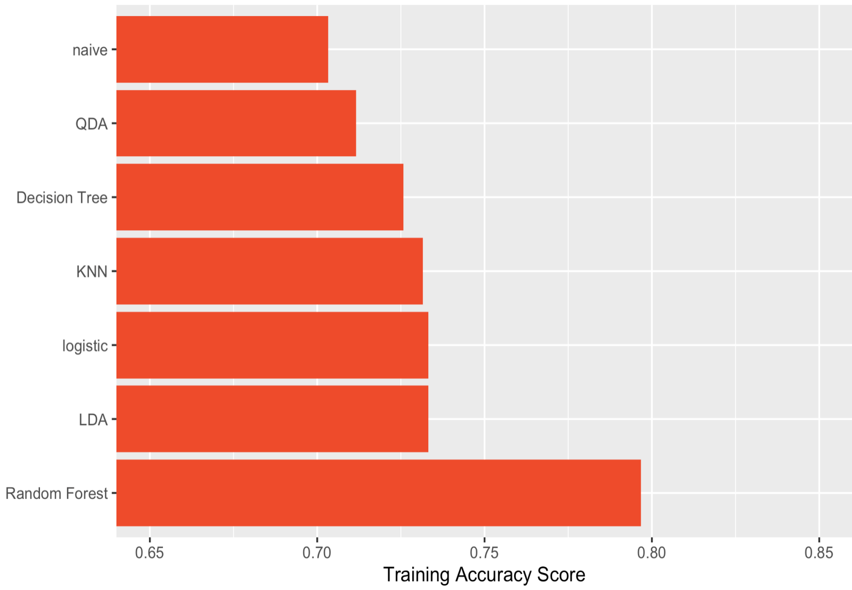

Figure 3: Training accuracy (sorted)

ROC is the most common way to visualize the trade-off between true positive rate and false positive rate when the threshold changes. ROC and AUC were used as the evaluation metrics for selecting the best model. The larger the AUC, the better the ROC. From the ROC curve, all six models have better ROC curves than the naive model, indicating that all six models were improved compared with the naive model. From figure 5, Logistic Regression has the highest AUC score, 0.813.

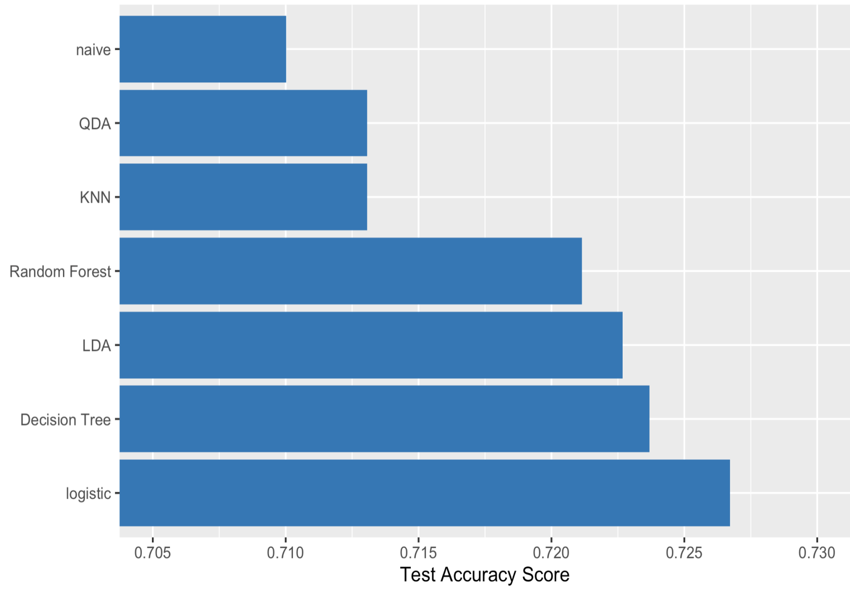

Figure 4: Test accuracy (sorted)

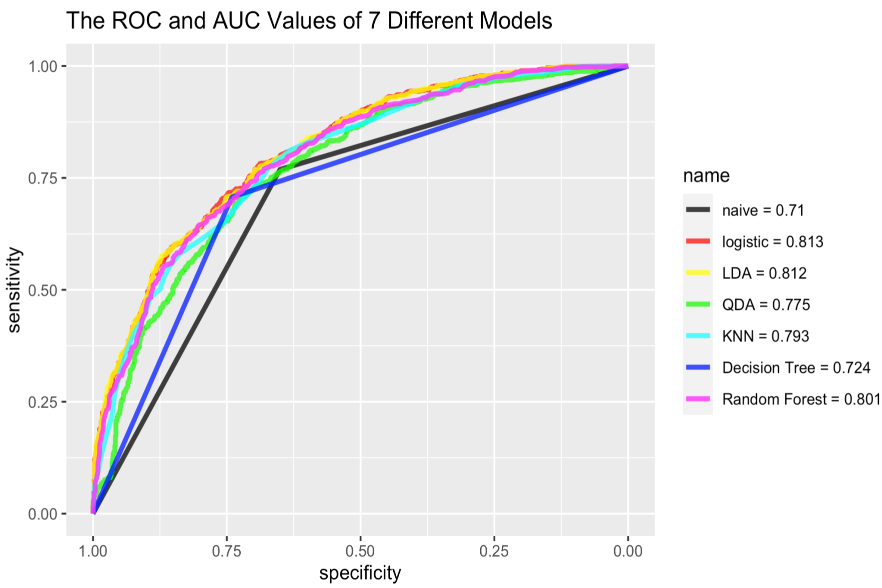

Figure 5: ROC curve plot

6 DISCUSSION

Based on the comparisons we made, we came up with the conclusion that **Logistic Regression** was the best model. It had the lowest test error, and the best ROC curve and AUC score. From the Logistic Regression model, we found that Gold difference, Dragon slaying, and Experience difference are the three most important features. Thus, if the players want to win the game, they should pay more attention to these three factors during the first 10 minutes. There are some further analyses that still needed to consider.

All six models had better ROC curves than the naive model, but the improvement was not very significant. There might be several reasons that need to be considered such as each model emphasizing different variables leading to the result of each fit not being that good. So we combined all the models into a new model in order to consider it comprehensively. The training and test errors were 0.2547 and 0.2723 respectively. The results did not change too much in the combined model, indicating the problem is not about the models themselves. It might be because some other factors were not considered like champion selection should be considered, or the first 10 minutes can not really determine the outcome of a match.

It is also possible that the models we used are not considering the interaction of each variables, so in the next step we may consider training a more complicated model such as SVM or CNN. More precise tuning methods are also in consideration for the improvement of current models.

A APPENDIX

A.1 DATA

The raw data set comes from Kaggle (link). It is an open sourced .csv file uploaded 2 years ago.

A.2 CODING

We have open sourced our coding section in GitHub. Please find the complete project in this repository.

`https://github.com/zoexinchen/MachineLearning_LOL_10min`.

In this repository you can find the raw data set, a reproducible .rmd file, a kniited pdf, a brief presentation video and the slides. You are welcome to email us for any comments and suggestions.

A.3 CONTRIBUTION

Xin Chen stated and defined the problem in the introduction, conducted part of the data preparation including detecting missing values, collinearity, and manipulation of the response, and performed models including, Logistic Regression, LDA, and QDA. For the final report, she was in charge of Introduction, Data and Experiment setup, and Results.

Ruiming Yu conducted part of the data preparation including splitting training, test, and cross-validation dataset, and performed models including KNN, Decision Tree, and Random Forest. For the final report, he was in charge of Related work, Methods, and Discussion.