Prediction of game results based on League of Legends

Jingwen Cong
U6502494

Report submitted for the course COMP8755: Individual project
Supervised by: Penny Kyburz
The Australian National University

June 2020

©Jingwen Cong 2020
Except where otherwise indicated, this report is my own original work.

Jingwen Cong
10 June 2020
Acknowledgments

First of all, I am very grateful to my mentor. Until this semester, due to COVID-19 and social isolation policies, all project processes have been converted to operate at home, and all meetings have been changed to online meetings. But my supervisor carefully and enthusiastically guides my project progress every week, answers my questions in a timely manner and puts forward many good suggestions.

I also want to thank my classmates. When I was accidental about some algorithms, they discussed the solution to the problem with me, exchanged ideas with me, and offered ideas for help.

At the same time, I am also very grateful to my parents. Due to the special epidemic situation, I often feel anxious and stressed during this time. When I was distressed by the project, they helped me to calm my emotions and comforted me to encourage me. Support me mentally to complete this project.
Abstract

The purpose of this project is to train a variety of machine learning algorithms and neural network models based on the existing massive League of Legends game data. And based on the data captured by the API at a certain point in the course of the game, the win rate of the two sides of the game at that moment is predicted. After the game, according to a series of data captured during the entire game, the win rate change chart of the two parties can be automatically generated during the game, so as to have a more intuitive understanding of the victory and defeat inflection points of the two teams in the game. It can also help the players and coach review the performance after the whole game.

This article will introduce the classic regression model, decision tree, GRNN neural network, follow-up forest, SVM vector machine and other prediction models used in the experiment, and describe the process of parameter selection in the experiment, and then compare and evaluate the performance of the prediction results of each prediction model. This project also includes a simple visual interface, which includes several models that perform well among all the models that have been tested, that is, the prediction model which is most suitable for the data of this project. Through this visual interface, the user can choose to use the API provided by Fist to extract the game play data of the game at a certain moment, so as to obtain the prediction result of the two sides at that moment. And after the game, I got a dynamic curve image about the change of win rate of both sides.
Contents

Acknowledgments............................................................................................................. iii
Abstract......................................................................................................................... v
Contents......................................................................................................................... vi
1. Introduction.................................................................................................................. 8
   1.1 Motivations............................................................................................................. 8
   1.2 Project Scope........................................................................................................ 9
   1.3 Report Outline...................................................................................................... 9
2. Background and Related work.................................................................................... 11
   2.1 Machine Learning in data mining........................................................................ 12
   2.2 Neural Network................................................................................................... 14
   2.3 N-fold cross-validation....................................................................................... 14
3. Experiment Environment............................................................................................ 11
   3.1 Experiment targeting........................................................................................... 11
   3.2 Experimental environment setup.......................................................................... 11
   3.3 Coding environment............................................................................................ 12
   3.4 Sklearn Library.................................................................................................. 12
4. Algorithms and Experiment Process........................................................................ 13
   4.1 Data Processing.................................................................................................... 13
   4.2 KNN classification.............................................................................................. 14
      4.2.3 Model evaluation.......................................................................................... 17
   4.3 Decision Tree....................................................................................................... 19
   4.4 Logistic Regression............................................................................................. 23
      4.4.2 Implement and experiment processes......................................................... 24
      4.4.3 Model evaluation........................................................................................ 28
   4.5 Support Vector Machines (SVM)....................................................................... 28
   4.6 Adaboost............................................................................................................. 31
   4.7 Naive Bayes....................................................................................................... 35
   4.8 GRNN................................................................................................................ 37
   4.9 Summary............................................................................................................. 39
5. Model Comparison.................................................................................................... 40
   5.1 Performance comparison.................................................................................... 40
   5.2 Summary............................................................................................................. 41
6. Applications................................................................................................................ 42
   6.1 Result Prediction based on the data of a certain time......................................... 42
   6.1 Winning rate changing line-graph........................................................................ 43
7. Conclusion and Future Works................................................................................... 45
   7.1 Conclusion.......................................................................................................... 45
7.2 Future Works..............................................................................................................45
Bibliography.....................................................................................................................47
Appendix 1.........................................................................................................................49
Chapter 1

Introduction

In recent years, e-sports has become one of the indispensable ways for people's daily entertainment. With the popularization of e-sports games, League of Legends, as a typical representative of confrontational video games, has also joined the ranks of international sports events. While holding regular international events every year, the League of Legends has also stepped onto the stage of the Asian Games and has become one of the projects of electronic sports performances. The use of machine learning and deep learning algorithms to assist in the prediction of the game situation, as well as to assist the coach in the analysis and review of the game, has also become a hot research topic.

1.1 Motivations

In some current League of Legends events, a predictive robot called Colonel KI is widely used. The prediction system is trained based on a large number of official data of the League of Legends international events. It can predict the outcome of the game in the role selection stage at the beginning of the game, and can also make results based on the economic conditions of the two games in the game process. However, because the data set that Colonel KI learns to understand the game data of professional games, and there is a certain gap between the situation of professional games and the match between most ordinary players, Colonel KI cannot be used in daily games and training games. Provide analytical assistance and forecasting. And after the game, Colonel KI was unable to provide intuitive changes in win rate for the rematch after the game. In addition to formal professional competitions, in many cases, ordinary players, even some high-end players, need data as an aid based on their own experience to make a review of their game situation, thereby enhancing their game awareness. And have better performance in the game.
1.2 Project Scope

Data processing and analysis: Part of the data set of this project is derived from the League of Legends data set downloaded on kaggle, and the other part is derived from the game data captured in the game match using the official League of Legends API provided by the fist game company. After that, data extraction technology was used to convert the extracted data into a usable data format. And when the data set is used for model training, some data processing techniques are used, so that the data set retains the highly correlated feature columns on the basis of available and tidy. The values are normalized and normalized to make the values and structure of the data set suitable for various machine learning and deep learning algorithms.

Supervised learning: In the training process, each training data set will correspond to a column, representing the game’s win or loss, so each data contains the label of the game’s winner, so each classification or regression model is used. During training, supervised learning is used.

Machine learning: Since the game only has two outcomes, that is, the prediction of the outcome of the game can be reduced to a non-zero or one binary classification problem. Therefore, you can use the classifier and vector machine to train the existing data set and predict the result of the game (the victory and loss of both parties). In this project, I will use traditional logistic regression algorithm, Adaboost adaptive classification algorithm, naive Bayes classification algorithm, KNN classification algorithm, decision tree classification algorithm and random forest classification algorithm, support vector machine based on known data Classification and prediction.

Deep learning: Since the purpose of this project is not only to make predictions on the results, but also to review the changes in the win rate of both sides of the normal game after the game, so only relying on the classifier cannot get the whole game. The winning percentage changes. This project will use the GRNN neural network with the advantages of extremely fast convergence and nonlinear approximation, to predict the win rate of both parties at specific time nodes in the game, and use the results as the basis for the game after the game.

Cross-validation method: Since the number of data sets used for training is not large, cross-validation method is used to evaluate the training performance of the model. While avoiding the phenomenon of over-fitting to a certain extent, we can mine as much useful information as possible from a limited data set.

1.3 Report Outline

The rest of this report is arranged as follows: In Chapter 2, I will introduce the background and related works of this project. In Chapter 3, I will introduce the experiment environment which set up in advance. In Chapter 4, all the
algorithms and the experiments processes of implementing these models and selection of parameters will be introduced. In Chapter 5, all the performance of the models used in this project will be integrated together and the comparison will be done. In Chapter 6, two real applications and interface designed for users will be listed and introduced. In Chapter 7, the conclusion and future works will be described.
Background and Related work

This project is based on artificial intelligence learning algorithms and data mining related fields, including machine learning algorithms, convolutional neural network algorithms, and widely used cross-validation methods. This chapter will introduce the machine learning algorithm used in this project, the algorithm and basic knowledge of neural network. And the application of this knowledge in data mining. After these, the cross-validation method applied in this experiment will be introduced. Finally, I will introduce the basis and method for adjusting the parameters of each algorithm, as well as the methods and guidelines for verifying which algorithms are most suitable for the data set used in this project.

First of all, the basic relations between AI algorithm, data mining, machine learning, and deep learning can be simply drawn as the following picture.

![Relations among AI, ML, DL and DM](image.png)
2.1 Machine Learning in data mining

There are many famous definitions of data mining which can be seen through the internet or related resource, one of the most famous saying is ‘A science that extracts useful information from large amounts of data or databases.’ (Hand, Mannila & Smyth, 2001). As well as machine learning is devoted to studying how to use experience to improve its performance through computational means, and experience refers to a large amount of data that is already available or collected (Mannila, 1996). Machine learning includes a series of algorithms that are feasible at a theoretical level, which are learning algorithms, and then apply the data set to these algorithms to generate a learning model. The trained learning model can be applied to a new data set to make predictions and judgments based on the data and features of the new data set (Mitchell, 1999). To a certain extent, data mining can be regarded as the application of machine learning combined with specific data sets in specific directions or disciplines. Machine learning is divided into supervised learning and unsupervised learning. In this project, the machine learning models used are all supervised learning. The work flow of supervised learning can be shown in the following diagram.
Figure 2.2 Work-flow of supervised learning

Under supervised learning, each set of training data has a clear identification or result. For example, in the data set of this project, each piece of data consists of a series of feature values and a label. The label is the win and loss of each side of each game. When establishing a prediction model, supervised learning establishes a learning process that compares the prediction results with the actual results of the "training data" and continuously adjusts the prediction model until the prediction results of the model reach an expected accuracy rate. As well as in this project, all the labels can be simply divided into ‘win’ or ‘loss’, so this can be regarded as a binary classification problem. So there are some classification
algorithm models can be used in this project, which are ‘Decision Tree’, ‘Naive Bayesian’, ‘Logistic Regression’, ‘Support Vector Machine’, ‘k-nearest neighbor’, ‘Neural Network’ and so on (Goldberg & Holland, 1988). The detailed knowledge of machine learning will be introduced in the next chapter.

2.2 Neural Network

Deep learning is a way or a path to machine learning. The motive is to establish and simulate the neural network of the human brain for analysis and learning. It mimics the mechanism of the human brain to interpret data (Specht, 1991). For example, it is connected according to a specific physical distance; while deep learning uses independent layers, connections, and data propagation directions. Convolutional neural network is the first true multi-layer structure learning algorithm. It uses spatial relative relationship to reduce the number of parameters to improve training performance, let the machine cognitive process proceed layer by layer, and gradually abstract, thereby greatly improving the accuracy and efficiency of recognition.

2.3 N-fold cross-validation

N-fold cross-validation always has two main uses: model evaluation and selection. Especially when the size of data set is not big enough, the cross-validation can avoid the over-fitting problems in some extent, as well as can maximize the use of limited data set (Fushiki, 2011). The data set is divided into a training set and a test set. The training set and the test set are divided in an N-fold cross way. Compared to the traditional model evaluation method (a fixed training set and a test set are divided), cross-validation The advantage is: to avoid problems caused by unreasonable data set division, such as overfitting the model on the training set, this overfitting may not be caused by the model, but because of unreasonable data set division. The situation is easy to appear when training models with small-scale data sets, so it is more advantageous to use cross-validation to evaluate models on small-scale data sets. In this project, considering about the size of the data set and other considerations, the fold of the validation is chosen to be 5. So the working flow of the 5-fold cross validation can be shown as the following diagram.
Figure 2.3 Work-flow of cross-validation
Experiment Environment

3.1 Experiment targeting

"League of Legends" (LOL for short) is a MOBA competitive online game developed by American fist game (Riot Games). There are hundreds of personalized heroes in the game. The experimental materials and experimental goals of this project are based on the ranking game of League of Legends. The ranking game consists of two teams, each of which contains five members. Before qualifying, each player can ban a hero character from appearing in this game (Kou & Gui, 2014). After the ban session, each player can choose one of all hero characters to participate in the game. In the whole process of the game, each player can obtain economics by killing creeps and public biological resources, or by killing the characters operated by the other player. The economy can be used to purchase weapons and equipment used in this game, when one party destroys the other teams’s base crystal, this team wins.

3.2 Experimental environment setup

3.2.1 Experiment set

The dataset which is used in this project mainly has two source: Kaggle website and dataset which is collected by using the Riot Games API.

3.2.2 Kaggle dataset

Founded in 2010, Kaggle is an online platform for data mining and prediction competitions, which includes a large number of databases for everyone to download and use. Some part of the dataset I used in this project is downloaded from Kaggle website: https://www.kaggle.com/datasnaek/league-of-legends This dataset includes over 50,000 ranked games from the game League of Legends,
3.2.3 Riot Game API

Here are many official API which aims to many games of Riot game company can be downloaded through Riot website. I downloaded the API for League of Legends rank games and collected the information through the rank history by user-id. The format of files caught by API is json which contains many aspects of information. In addition, the user id is caught through the game history of my friend and the teammates and opposite players he met in the ranking games. For further verification, more data can be scraped by many 3rd part LoL websites.

3.3 Coding environment

Python is a good language for data processing and analysis, including implementing many machine learning and deep learning methods. As well as there are many packages which help a lot to realize many useful training and study algorithms, so the code part of the whole project is mostly implemented by python. In this project, Jupiter is very convenient in visualizing the names and characteristics of the rows and columns of the dataset, during the procession on the number and the structure of the dataset. So the main function which implement the study algorithms and prediction models was completed on Jupiter Notebook.

3.4 Sklearn Library

Sklearn is a very powerful machine learning library provided by a third-party Python. It includes every aspects from data preprocessing to training models. Using scikit-learn in actual project can greatly save the time we write code and reduce the amount of our code, so that we have more time to analyze the data distribution, adjust the model and modify the parameters. In this project, after basic data processing, some of the prediction algorithms are based on scikit-learn library.
Chapter 4

Algorithms and Experiment Process

By observing the structure of the data set and combining with the machine learning knowledge mentioned in the previous chapter, we can see that this project is a binary classification problem. Because the labels can simply be divided into 0 and 1, which mean team 1 win the game and team 2 win the game. So this project is about classification methods and algorithms based on the machine learning. This project is consist of four processes, the work flow is shown in Figure1. Firstly, after gathered the dataset through the methods mentioned in the last chapter, the dataset is processed by selecting useful attributes and value normalization. After that, the whole dataset is divided into training and testing parts. Secondly, the training dataset which has been processed are used to train many machine learning and deep learning models. Thirdly, after analyzing and comparing the performances of different prediction models, three best models are selected and saved locally for further prediction. Finally, a simple visual interface is used for users to select the data which was caught by official API provided by Riot Game website, and get the prediction result.

4.1 Data Processing

First of all, the data captured by the official API is a json file, and contains many values and features that are not used in this experiment, so first of all, you need to extract the relevant files in the json file and convert them into csv format to apply to the experiment. And the rest of the data set is downloaded from kaggle website. After downloading the data set, the data caught by API and downloaded data can be integrated into one data set, and the structure and distribution of the value can be viewed easily. The dataset is consist of 51490 rows and 61 columns, 49 columns have integer values and 12 columns have ID values. After getting a general view on this dataset, the processing of dataset can begin with cleaning duplication data and deleting bad-data. After doing some research, the games whose last-times are shorter than 10 minutes are always finished by some uncommon reasons, such as the phenomenon of unequal numbers of two sides caused by players hanging up, or a early surrender caused by some players’ sudden situation, or simply the disparate technology gap between two sides. All of those situations are caused by uncommon reasons or the mismatch of the ranking system, so they should not be under consideration. So the rows whose game last-time is less than 10 minutes are deleted. After that, when trying to fill in missing values, it is found that there is no missing value in this data set. So it is a clean dataset in some extend. For transferring the label
line into the correct format which can be identified and directly used in the machine learning models, the column of ‘winner’ should be transferred from integer 1 and 2 to binary format 0 and 1. After those operations, there are 50286 rows remains which can be used in this project.

4.2 KNN classification

4.2.1 Introduction of KNN

KNN classifies work by measuring the distance between different feature values. The idea is: if most of the k most similar samples in the feature space (that is, the nearest neighbors in the feature space) belong to a certain category, the sample also belongs to this category, where K is usually not greater than An integer of 20. In the KNN algorithm, the selected neighbors are all objects that have been correctly classified. This method determines the category to which the samples to be classified belong based on the category of the nearest sample or samples in the decision of classification (Soucy & Mineau, 2001). Here is a simple example to illustrate: As shown in the figure below, which class is the green circle to be determined, is it a red triangle or a blue square? If K=3, because the proportion of red triangles is 2/3, the green circle will be assigned to the category of red triangles. If K=5, since the proportion of blue squares is 3/5, the green circle will be assigned to blue four Square type. This also shows that the result of the KNN algorithm depends largely on the choice of K.

![Figure 4.1 KNN work principle](image)

In KNN, the distance between objects is calculated as a non-similarity index between objects to avoid the matching problem between objects. Here, the distance is generally Euclidean distance or Manhattan distance. Euclidean distance is:

\[ d(x,y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2} \]
And Manhattan distance is:

\[ d(x, y) = \sqrt{\sum_{k=1}^{n} |x_k - y_k|} \]

To summarize the idea of the KNN algorithm: when the data and labels in the training set are known, enter the test data, compare the characteristics of the test data with the corresponding features in the training set, and find the most similar top \( K \) in the training set. Data, the category corresponding to the test data is the category with the most occurrences among the \( K \) data.

### 4.2.2 Implement and experiment processes

When we use KNN classification, we can firstly call KNN classification method provided in the sklearn library mentioned in the experiment environment chapter. In the KNN algorithm, the choice of \( k \) value has a significant impact on the accuracy of the results and whether there is any over or under fitting. When \( K \) is too small, it is equivalent to predicting with training examples in a smaller field. The "learning" approximation error will be reduced. Only training examples that are close to or similar to the input examples will have an effect on the prediction results, and at the same time bring problems. It is the estimation error of "learning" will increase, in other words, the reduction of the \( K \) value means that the overall model becomes complicated and prone to over-fitting. In opposite, when \( K \) is too large, it is equivalent to predicting with training examples in a larger scale. Its advantage is that it can reduce the estimation mistakes of learning, but the disadvantage is that the approximate error of learning will increase. At this time, the training examples that are far away (not similar) from the input examples will also act on the predictor, making the prediction wrong, and the increase in the \( K \) value means that the overall model becomes simple, so the under-fitting may occur in this condition. So the selection of \( k \) value is pretty significant for the accuracy of the prediction. So I try the \( K \) value from 1 to 39, test the model and observe the changes of the prediction accuracy.
We can see from the line-graph above that, the accuracy increases firstly, and then decrease after the peak. When k=5, the accuracy reaches the highest point at 0.977, so the most accurate k value we use in KNN classification is 5.

However, the performance of a model can not only depended on the accuracy during the training process, but also the fitting states, which contains if there exists some over-fitting or under-fitting problems. The classification can be evaluated by cross validation score, and the learning curve is to draw the correct rate of the training set and cross-validation when drawing different training set sizes (Spence, 1981). We can see the performance of the model on the new data, and it can be observed to judge whether the model has high variance or high deviation, and whether increasing the size of the training set may lead to over-fitting. As the content mentioned previously, larger k value may lead to less generalization ability and more over-fitting problems, so we draw the learning curve graphs of the model when k=4 and k=5 and see if the the model exists some over-fitting problems when k=5 and if the decrease of k value can solve the over-fitting problem. In the cross validation, the performance of this KNN classification models with different k values can be drawn as following graph.
The red line represents the accuracy changes along with the size increasing of the training set, and the green line represent the accuracy changes along with the size increasing of the validation data set. So we can see that, the accuracy can reach around 0.9815 when the training set reaches the highest point, and the highest accuracy of the validation set is around 0.975. The accuracy of the training set is still higher than the accuracy of test set, at this point. So there exists some over-fitting problems. We can see from the graphs that, when k=4, the cross validation score is lower than k=5, and still exists some over-fitting problems. So 5 seems like the best choice of k value.

4.2.3 Model evaluation

4.2.3.1 Confusion matrix

In this stage, we will use confusion matrix to express accuracy evaluation. In the field of machine learning, the confusion matrix (confusion matrix), also known as the possibility table or error matrix. It is a specific matrix used for supervised learning. Each column represents the predicted value, and each row represents the actual category. The name comes from the fact that it can easily indicate whether there is confusion among multiple categories (that is, one class is predicted to be another class). In predictive analysis, a confusion table (sometimes called a confusion matrix) is a two-row, two-column table consisting of false positives, false negatives, true positives, and true negatives, and the structure of the confusion table and the meanings of these four values are shown in the following table (Visa, Ralescu & Van, 2011).
And the confusion table can be got based on the k=5 KNN classification.

<table>
<thead>
<tr>
<th>True positive (TP)</th>
<th>False negative (FN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predict positive classes</td>
<td>Predict positive class</td>
</tr>
<tr>
<td>as positive class numbers</td>
<td>as negative class number</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>False positive (FP)</td>
<td>True negative (TN)</td>
</tr>
<tr>
<td>Predict negative class</td>
<td>Predict negative class</td>
</tr>
<tr>
<td>as positive class number</td>
<td>as negative class number</td>
</tr>
</tbody>
</table>

Figure 4.4 Confusion matrix

<table>
<thead>
<tr>
<th>TP</th>
<th>FN</th>
</tr>
</thead>
<tbody>
<tr>
<td>4871</td>
<td>117</td>
</tr>
<tr>
<td>111</td>
<td>4873</td>
</tr>
</tbody>
</table>

Table 4.1 Confusion matrix of KNN

4.2.3.2 Accuracy, recall, precision and f_score

For binary prediction model, there are four indexes which can be used for evaluating the performance of the model and the algorithm. They are accuracy, recall, precision and f_score.

**Accuracy**: Although the accuracy rate can determine the overall accuracy rate, it cannot be used as a good indicator to measure the results when the sample is not balanced. For example, in the sample set, there are 90 positive samples and 10 negative samples. The samples are severely unbalanced. For this situation, we only need to predict all samples as positive samples, and we can get 90% accuracy, but it is completely meaningless. The formula of Accuracy is:

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + FN}
\]

**Precision**: For the prediction result, its meaning is the probability of actually being a positive sample among all the samples that are predicted to be positive. The accuracy rate and accuracy rate are somewhat similar, but they are two completely different concepts. The accuracy rate represents the accuracy of the prediction in the positive sample results, and the accuracy rate represents the overall accuracy of the prediction, including positive and negative samples. The formula of Precision is:

\[
\text{Precision} = \frac{TP}{TP + FP}
\]

**Recall**: Recall is for the original sample, and its meaning is the probability of being predicted as a positive sample in the actual positive sample. The accuracy
rate and the recall rate affect each other. Ideally, the two must be high, but the actual situation is that the two "restrict" each other: the pursuit of a high accuracy rate means a low recall rate; the pursuit of a high recall rate usually affects accuracy rate. The formula of Recall is:

\[ \text{Recall} = \frac{TP}{TP + FN} \]

**F-score**: Generally speaking, the accuracy rate and the recall rate are negatively correlated. One is high and the other is low. If both are low, there must be a problem. So there is a contradiction between accuracy and recall rate. Here, F-Score is introduced as a comprehensive indicator, in order to balance the impact of accuracy rate and recall rate, a classifier is evaluated more comprehensively. F-Score is the harmonic average of precision and recall. The formula of F-score is (Junker, Hoch & Dengel, 1999):

\[ F = \frac{2PR}{P + R} = \frac{2TP}{2TP + FP + FN} \]

And the results of these four features can be calculated from the confusion table, and can be shown in the following table.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Nearest Neighbours</td>
<td>0.97714</td>
<td>0.97654</td>
<td>0.97772</td>
<td>0.97713</td>
</tr>
</tbody>
</table>

Table 4.2 Accuracy, Recall, Precision and F-score

### 4.3 Decision Tree

#### 4.3.1 Introduction of decision tree

A decision tree is a tree structure in which each internal node represents a judgment on an attribute, each branch represents the output of a judgment result, and finally each leaf node represents a classification result of decision tree. Therefore, the generation of the decision tree is mainly divided into the following two steps. These two steps are usually achieved by learning samples that already know the classification results: 1. Node splitting: Generally, when the attribute represented by a node cannot be judged, the node is divided into 2 child nodes (if it is not a binary tree, it will be divided into n child nodes) 2. Determination of the threshold: Choose an appropriate threshold to minimize the classification error rate (Training Error) (Friedl & Brodley, 1997). And the decision tree principles can be simply described as the following four steps:
Step 1: Get the entropy of each feature and compare it with the original data entropy to get the information gain of the feature.
Step 2: Select the optimal feature with the largest information gain from it, and take it as the current node.
Step 3: Eliminate the current node and continue to repeat steps 1 and 2 recursively.
Step 4: Two conditions end 3 recursion. 1: The target variable under the current node is unique. 2: All features are looped.

The 4 points above are the most critical points in creating a decision tree, which is also its basic principle. Around these four points we can create a simple decision tree.

As the working principle mentioned in the previous content, the decision tree has split in each layer based on the different eigenvalues. The time of split is equal to the max_depth of the decision tree. And the split of the decision tree in this project can be drawn as a pdf diagram.

This graph can directly shows the basis and working principle of each split. So the max_depth of the decision tree is pretty significant to ensure the performance prediction result of the model in both good accuracy and fitting status.

### 4.3.2 Implement and experiment processes

Taking a max_depth of 3 as an example, each split of the decision tree can obtain the following visual results.

![Decision Tree](image)

**Figure 4.5 Decision tree (depth=3) of LOL data set**

This graph can directly shows the basis and working principle of each split. So the choice of max_depth of the decision tree is pretty significant to ensure the performance prediction result of the model in both good accuracy and fitting status.

In this section, I’ll combine the accuracy and fitting situation together to get the best parameter. Firstly, we can change the max_depth of the layers from 4 to 11,
and plot the diagram of the accuracy changing along with the max_depth changes. It can be shown in the following line graph, in this line graph, the x axis means the max_depth, and the y means the accuracy of train set.

![Accuracy Graph](image)

**Figure 4.6 Accuracy changing increasing with depth**

It can be seen from the graph that when the max_depth of the decision tree is 9, the train set accuracy is the highest. However, the choice of max_depth can not only depend on the accuracy of the train set because the over-fitting will become more serious as the max_depth increases, so we should plot the learning curve of the train set and validation set together to analysis the fitting state. The following diagrams are the learning curve of the decision tree when max_depth=4 to 9.
From the diagrams above, we can see that, when the max_depth of decision tree is 4 and 5, the accuracy of both train set and test set keep decreasing when the size of train set improves from 22000 to the maximum number, so when max_depth equals to 4 and 5, this model cannot accurately predict the known data and the unknown, which belongs to high deviation. In this case, the model is likely to be under-fitting. And when the max_depth is 6, the accuracy of test set is below 0.976, the score is lower than the rest of the models, and still has over-fitting problem. When the max_depth of the decision tree are 7, 8, and 9, all of the scores of the cross validation set are around 0.9775, and the modes are all have some over-fitting phenomenon, and the most serious over-fitting problem is when max_depth of decision tree equals to 9. We should choose the model which is less over-fitting and less complexity, so it seems that when max_depth=7, this model performance is the best.
4.3.3 Model evaluation

So we can get the confusion value of the decision tree model whose depth is 7 in the following table.

<table>
<thead>
<tr>
<th>TP=4895</th>
<th>FN=93</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP=118</td>
<td>TN=4866</td>
</tr>
</tbody>
</table>

Table 4.3 Confusion matrix of decision tree

And the Accuracy, Precision, Recall and F-score ca also be calculated based on the confusion matrix:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree (max_depth=7)</td>
<td>0.97884</td>
<td>0.98136</td>
<td>0.97646</td>
<td>0.97890</td>
</tr>
</tbody>
</table>

Table 4.4 Accuracy, Recall, Precision and F-score of decision tree

4.4 Logistic Regression

4.4.1 Introduction of decision tree

Logistic regression is actually a classification algorithm, but for historical reasons it is called logistic regression. In the classification problem, the variable y to be predicted is a discrete value. Logistic regression is currently the most popular and widely used learning algorithm. The hypothesis algorithm model of logistic regression is:

\[ h_\theta(x) = g(\theta^T x) \]

In this function, X represents the feature vector, g represents the logistic function. As well as the outcome in logistic regression classification is restricted between 0 to 1, so a hypothesis function that can satisfy the prediction value between 0 and 1 should be thought out. It means that the outcome of the function ‘g’ should between 0 to 1. A Sigmoid function can be used to meet this requirement.

\[ g(z) = \frac{1}{1+e^{-z}} \]

And the image of this function (S-Curve) can be shown in the following image.
So the logistic regression model can be created by combining them together.

\[
h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}
\]

It is obviously that, the value of \( h \) is between 0 to 1, but not a binary value, so the usage of \( h_\theta(x) \) is to calculate a value which represents the estimated possibility that output value =1 based on the selected parameters. For example, if the value of \( h \) is 0.9, which means the possibility of \( y \) is a positive class is 0.9. In the binary problems, we can regard those whose possibility bigger than 0.5 belongs to 1 class, and the rest belong to 0 class. The meaning of this function can be represented by the following expression.

\[
h_\theta(x) = P(y = 1 | x; \theta)
\]

And the cost function of logistic regression is also different from the others.

When the truth value is 1 class, the first function us used to represent the cost function, when the truth value is 0 class, the second function is used to represent the cost function. The combination of the above two formulas is the following function, plus the regularization term (Boyd & Vandenberghe, 2004).

\[
J(\theta) = \left[ -\frac{1}{m} \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log 1 - h_\theta(x^{(i)}) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2
\]

Then use the gradient algorithm to calculate the minimize point, which is the result of the expected model.

### 4.4.2 Implement and experiment processes

From the content above, we can see that one of the most significant process of this algorithm is the minimization of the cost function. It can be regarded as a kind of optimization problem. In Regression algorithm provided by ski-learn, there are five minimization method which can be used to find the optimization solution, which are:
i. Liblinear: The coordinate axis descent method is used to iteratively optimize the loss function.

ii. lbfgs: A kind of quasi-Newton method, the first derivative (gradient) of the function is used to approximate the recursive Hessian matrix.

iii. Newton-cg: It is also a kind of Newton's method family which uses the second derivative matrix of the loss function, that is, the Hessian matrix, to iteratively optimize the loss function.

iv. Sag: Stochastic average gradient descent is a variation of the gradient descent method. The difference from the ordinary gradient descent method is that each iteration uses only a part of the sample to calculate the gradient, which is suitable when there are many sample data.

v. Saga: A variant of a linear convergent stochastic optimization algorithm. In general, liblinear is suitable for small data sets, while sag and saga are suitable for large data sets because it is faster. In my project, I try each of these five methods, and the following accuracy diagram can be plotted.

![Accuracy of different optimization algorithms](image)

**Figure 4.9 Accuracy of different optimization algorithm**

It is obvious that when the parameter is selected to be ‘liblinear’, the model has the best performance based on training set. And another parameter, C, represents for the reciprocal of the regularization coefficient $\lambda$, that is, the smaller the value, the stronger the degree of regularization. In opposite, when C is larger, it means the higher accuracy, lower fault tolerance rate, but worse generalization ability. It is a float type, and must be a positive floating point number. Because the changes of C value is always based on 10-n, which is very significant change, so I choose to find the scope of c which may contain the best selection of C value based on the learning curves.
We can see from the graphs that when \( c = 0.01 \), the model has the best fitting but validation set score is much smaller than the other two models. When \( C = 1 \), the model has the highest validation score, however, the fitting state is the worst.

Figure 4.10 Learning curves of different logistic regression model
among those models which based on 3 selection of C values. So we can observe from the graphs than maybe the best choice of C value is between 0.1 to 1. So I will try from 0.01 to 1 and select the most suitable value for c. The changes of the accuracy can be drawn as the following diagram.

![Figure 4.11 Accuracy line–graph changes](image)

We can see from the line graph that, when c=0.26, the accuracy is the highest, and we can generate a learning curve graph based on the model whose C value=0.26.

![Figure 4.12 Learning curve of selected line–graph](image)

We can observe from the graph that, when C=0.26, the accuracy is not bad, and there is almost no over-fitting problems when the training set achieves the highest point. So 0.26 seems like the best choice for parameter C.
4.4.3 Model evaluation

The confusion matrix can be calculated by the methods in previous content:

<table>
<thead>
<tr>
<th></th>
<th>TP=4842</th>
<th>FN=146</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP=113</td>
<td></td>
<td>TN=4871</td>
</tr>
</tbody>
</table>

Figure 4.10 Confusion matrix of logistic regression

And the accuracy, recall, precision and f_score can also be calculated based on the confusion matrix:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>0.96791</td>
<td>0.96552</td>
<td>0.97019</td>
<td>0.96785</td>
</tr>
</tbody>
</table>

Figure 4.11 Accuracy, Recall, Precision, F–score of logistic regression

4.5 Support Vector Machines (SVM)

4.5.1 Introduction of SVM

Support vector machines (SVM) is a binary classification model. Its basic model is a linear classifier with the largest interval defined in the feature space. The largest interval makes it different from the perceptron; SVM also includes kernel skills, which makes it a substantially nonlinear classifier. The learning strategy of SVM is to maximize the interval, which can be formalized as a problem of solving convex quadratic programming, and is also equivalent to the problem of minimizing the regularized hinge loss function. The learning algorithm of SVM is the optimal algorithm for solving convex quadratic programming. The basic idea of SVM learning is to solve the separation hyper-plane that can correctly divide the training data set and has the largest geometric interval. For linearly separable data sets, there are an infinite number of such hyper-planes (that is, perceptrons), but the separation hyper-plane with the largest geometric interval is unique (Joachims, 1999).

4.5.2 Implement and experiment processes

We can get from the previous content that the choice of C value is pretty significant for the performance of the model. Same as the C value of Logistic Regression, it also means the penalty coefficient of objects. The accuracy increase with the increasing of c value, however, the generalization ability keeps decreasing. So we should combine the accuracy graph and the learning curves
together to select the best C value with both acceptable fitting states and validation score.

![Learning curves of SVM](image)

**Figure 4.13 Learning curves of SVM**

Same as the parameter selection method mentioned in the last algorithm, we can similarly get the scope of the best C is between 0.1 to 1, and then the learning
curve can be drawn based on this model.

We can see from the graph that, when the C value equals to 0.7 and 0.9, the accuracy can achieve the highest value. And the lower C value means the less over-fitting problem, so we choose 0.7 as the best choice of C. For non-linear situations, the processing method of SVM is to choose a kernel function k and to solve the problem of linear inseparability in the original space by mapping the data to a high-dimensional space. In the case of linear inseparability, the support vector machine first completes the calculation in the low-dimensional space, and then maps the input space to the high-dimensional feature space through the kernel function, and finally constructs the optimal separation hyperplane in the high-dimensional feature space, thereby putting Non-linear data on the plane itself is not easy to separate. As well there are also some kernels which are provided and can be selected in this model, so the performance of each kernel can still be observed through the following bar graph. We can see that the ‘rbf’ performs the best in this database.
4.5.3 Model evaluation

The confusion matrix can be calculated based on the SVM model:

<table>
<thead>
<tr>
<th></th>
<th>TP = 4884</th>
<th>FN = 104</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP = 101</td>
<td></td>
<td>TN = 4883</td>
</tr>
</tbody>
</table>

Figure 4.12 Confusion matrix of SVM

As well as the accuracy, precision, recall and f-score can also be calculated based on the confusion matrix:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>0.97954</td>
<td>0.9795</td>
<td>0.97955</td>
<td>0.97955</td>
</tr>
</tbody>
</table>

Table 4.13 Accuracy, Recall, Precision and F-score of SVM

4.6 Adaboost

4.6.1 Introduction of decision tree

Adaboost is an iterative algorithm whose core idea is to train different classifiers (weak classifiers) against the same training set, and then combine these weak classifiers to form a stronger final classifier (strong classifier). It determines the weight of each sample based on the correct classification of each sample in each training set and the accuracy of the previous overall classification. The new data set with modified weights is sent to the lower classifier for training, and finally the classifier obtained by each training is finally fused as the final decision classifier. This algorithm is actually a simple weak classification algorithm promotion process. This process can improve the classification ability of data through continuous training.
From that graph, we can see that the whole processes can be simply described as the following steps:

First obtain the first weak classifier by learning N training samples. Combine the wrong sample with other new data to form a new N training sample, and get a second weak classifier by learning this sample. Add the samples that are wrong to 1 and 2 together with other new samples to form another new N training samples, and get a third weak classifier by learning this sample. In the end, after the enhanced strong classifier, the category to which a certain data is divided should pass the majority vote (Khammari, Nashashibi, Abramson, & Laurgeau, 2005).

4.6.2 Implement and experiment processes

Firstly, the Adaboost classifier needs a base classifier as the choice of the weak classifier, and in this process we choose decision tree as the base classifier. And the n_estimators parameter of the classifier means the base classifier lift (cycle) times (Shen & Bai, 2004), which is the iteration time, and the default value is 50. When the n_estimators is too large, the model may have over-fitting problem, however if the n_estimators is too small, the model may have some under-fitting problems. So we should still choose the value of n_estimators based on the accuracy line graph and learning curves.
Figure 4.17 Accuracy changing line-graph based on iteration times

Firstly, from the line graph above, we can get the information that, when the n_estimators=40 and 100, the model have the highest accuracy. Then we can check if the model has some over-fitting problems when n_estimator is under these two values.

Figure 4.18 Learning Curves of Adaboost with different n_estimators
We can see from the learning curves that, the validation score of these two adaboost is pretty similar to each other, both of them are acceptable, however, the over-fitting problem when n_estimators=100 is more serious than the Adaboost model whose n_estimators=40. So in this stage, we can get the conclusion that 40 is the best selection of n_estimators for this project and data set. learning_rate is also another parameter of this algorithm model, and the learning rate, representing the gradient convergence speed, the default is 1, if it is too large, it is easy to miss the optimal value, if it is too small, the convergence speed will be very slow; this value needs to be weighed against n_estimators, when the number of classifier iterations is relatively high When there is less, the learning rate can be smaller, when the number of iterations is more, the learning rate can be appropriately enlarged. So the accuracy line graph can also be used to select the best value of learning rate. And from the graph, we can get that when learning_rate=1, the accuracy of the model is the highest.

![Figure 4.19 Accuracy changes along with the learning rate](image)

### 4.6.3 Model evaluation

The confusion matrix of this model can be generated based on this model, it can be shown in the following table:

<table>
<thead>
<tr>
<th></th>
<th>TP</th>
<th>FN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4884</td>
<td>104</td>
</tr>
<tr>
<td>FP = 91</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TN = 4893</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.14 Confusion matrix of Adaboost

And the accuracy, recall, precision and f-score can also be calculated based on the confusion matrix:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaboost</td>
<td>0.98045</td>
<td>0.97915</td>
<td>0.98171</td>
<td>0.98043</td>
</tr>
</tbody>
</table>

Table 4.15 Accuracy, Recall, Precision and F-score of Adaboost
4.7 Naive Bayes

4.7.1 Introduction of Naive Bayes

The principle of the algorithm is based on the prior probability $P(Y)$, using Bayes' formula to calculate the posterior probability $P(Y/X)$ (the probability that the object belongs to a certain class), and select the class with the largest posterior probability as the Object class. Naive Bayes always contains three main methods, they are Gaussian Naive Bayes, Polynomial Distribution Bayes and Bernoulli Naive Bayes. These three classification methods are response to three different types of data distributions (Zhang, 2004).

**Gaussian Naive Bayes**: This method is suitable for the data set whose distribution is normal distribution.

**Bernoulli Naive Bayes**: This method is suitable for the data set whose distribution is Bernoulli Naive Bayes. Bernoulli distribution is also known as "zero-one distribution" and "two-point distribution" (that is, the result is either 0 or 1). It is a special case of the binomial distribution. The reason why the binomial distribution is special is because the binomial distribution It is the probability distribution of multiple Bernoulli experiments.

**Polynomial Naive Bayes**: This method is suitable for the data set whose distribution follows Multinomial Distribution. It is a generalization of the binomial distribution. However, the polynomial distribution means that there are multiple random result values(the result of shaking the dice e.g.).

4.7.2 Implement and experiment processes

So from the above content, it seems that the Bernoulli is the most suitable algorithm for the data set used in this project, because the result(label) is either team1 win or team 2 win. So firstly I tried this model and find the accuracy based on this model is 0.947 only, which is lower than other algorithms. The learning curve of this model can be shown below.
The diagram above shows that the score of this model is pretty low, so it seems not the most suitable model which can be used in this data set. Then because the polynomial is a generalization of the binomial distribution, so we can simply know that the result still can not be good. Then Gaussian can be tried to see if the score can be improved. Gaussian can be used directly without any parameters so it is pretty convenient to use. And the learning curve can also be generated based on Gaussian algorithm.

We can see that the score of this algorithm is much better than last one, so we will calculate the confusion matrix based on this model.
4.7.3 Model evaluation

The confusion matrix can be calculated based on this model, and the result can be shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>TP=4767</th>
<th>FN=221</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FP=210</td>
<td>TN=4774</td>
</tr>
</tbody>
</table>

Table 4.15 Confusion matrix of Gaussian Naive Bayes

And the validation features can also be calculated based on the confusion matrix. They can be shown in the following table.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>0.95678</td>
<td>0.95569</td>
<td>0.95781</td>
<td>0.95675</td>
</tr>
</tbody>
</table>

Table 4.16 Confusion matrix of Gaussian Naive Bayes

4.8 GRNN

4.8.1 Introduction of GRNN

The generalized regression neural network is a kind of radial basis neural network. GRNN has a strong nonlinear mapping ability and learning speed. The network finally converges to the optimized regression with a large sample size aggregation. When the sample data is small, the prediction effect is very good. The network can also handle unstable data. The working flow of GRNN can be shown as the following graph.
**Input layer:** In which, the number of neurons of input layer equals to the dimension of the input vector in the sample. Each neuron is a simple distribution unit that directly transfers input variables to the pattern layer.

**Pattern layer:** The number of neurons of the pattern layer equals to the number of the study samples. Each neuron response to different sample. The transfer function of the pattern layer is:

\[ \text{Gauss}(\text{tex}_i - \text{trx}_j) = e^{-\|\text{tex}_i - \text{trx}_j\|^2/2\delta^2} \]

**Summation layer:** The number of the nodes equals to the dimension of the sample plus one (k+1). The output of this layer can be divided into two parts, the output of the first node is the arithmetic sum of the output of the mode layer, and the output of the remaining k nodes is the weighted sum of the output of the mode layer. There are both some advantages and disadvantages of GRNN algorithm. The convergence speed is quiet fast because no model parameters need to be trained. And another advantage is nonlinear approximation, because it is based on radial basis network, with good nonlinear approximation performance. However, the calculation and space complexity is pretty high, because each test sample should be computed with all the train samples.
4.8.2 Implement, experiment processes and model evaluation

Because in this algorithm, no parameter is needed to be selected, so we can get the validation result of this model directly without comparing models with different parameters and also do not need to generating learning curve.

The confusion matrix of this model can be shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>TP=4846</th>
<th>FN=142</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP=118</td>
<td>TN=486</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.17 Confusion matrix of GRNN

And the validation values can also be calculated based on the confusion matrix.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRNN</td>
<td>0.97393</td>
<td>0.97153</td>
<td>0.97623</td>
<td>0.97387</td>
</tr>
</tbody>
</table>

Table 4.18 Confusion matrix of GRNN

4.9 Summary

This chapter has introduced 7 algorithm models used in this project, and the performances of these models. And the best models and comparison among the models will be introduced in the next chapter.
Chapter 5

**Model Comparison**

This chapter will integrate the performance feature values of each algorithm and select the model that is most suitable for the project data through comparison.

### 5.1 Performance comparison

From the previous above, we can put the value of these four features together.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Recall</th>
<th>Precision</th>
<th>F-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Nearest Neighbours</td>
<td>0.97714</td>
<td>0.97654</td>
<td>0.97772</td>
<td>0.97713</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.97884</td>
<td>0.98136</td>
<td>0.97646</td>
<td>0.97890</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.96791</td>
<td>0.96552</td>
<td>0.97019</td>
<td>0.96785</td>
</tr>
<tr>
<td>Adaboost</td>
<td>0.98045</td>
<td>0.97915</td>
<td>0.98171</td>
<td>0.98043</td>
</tr>
<tr>
<td>Support Vector Machines</td>
<td>0.97954</td>
<td>0.97955</td>
<td>0.97955</td>
<td>0.97955</td>
</tr>
<tr>
<td>GRNN</td>
<td>0.97393</td>
<td>0.97153</td>
<td>0.97623</td>
<td>0.97387</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>0.95678</td>
<td>0.95569</td>
<td>0.95781</td>
<td>0.95675</td>
</tr>
</tbody>
</table>

Table 5.1 Accuracy, Recall, Precision and F-score of all models
And the bar graph can be generated according to this table.

![Validation features](image)

Figure 5.1 Accuracy, Recall, Precision and F-score of all models

We can know from the previous content that all of these four features (accuracy, recall, precision, f-score) can reflect the performance of a prediction model, so higher the average value of these features, better the performance of the model will be. So from the bar graph and the table we can see that, Adaboost and SVM models perform better than the rest of algorithms.

### 5.2 Summary

From this chapter, we compare the performance of the algorithms with each other based on accuracy, recall, precision and f-score values, and select the best performance models which are SVM and Adaboost to use in the future application.
Chapter 6

Applications

This chapter introduces the applications of the selected models, as well as GRNN model is still used in the application because this is the only neural network model which is used in this project, and we can compare the result of neural network and the other machine learning models in the real use.

6.1 Result Prediction based on the data of a certain time

These prediction models are applied to the prediction of game results based on game data at a certain moment. This includes an interface that allows users to select data and prediction models and get prediction results. Firstly, the user can select the game data file which is caught by official API from the files.

Figure 6.1 Interface screen–shot 1
Next, the user can click the one of the three model listed in the interface to select one model to do the prediction based on the selection file. Then the user can get the predicted result by the selected model like what is shown in the diagrams below.

6.1 Winning rate changing line-graph

Through the whole game, API can catch the game data every few minutes, and storage them in a certain path. Then a series of data will be put into the adaboost model, and get a series of winning rate prediction results. Then the winning rate changes result can be plotted to be a line graph. This is an example graph which is generated based on the data caught every one minute.
And we can obviously know that, at each cross point, it means the changes of the game situation. And this graph can help the coach and players review the whole game, and know at which point, which decision or behavior changed the situation of the game.
Conclusion and Future Works

7.1 Conclusion

In this project, I first processed the collected data, using some data processing, extraction and integration techniques to make the data set suitable for the next algorithm. Then some machine learning algorithms are used, and for the integrated data set, combined with the results of cross-validation and learning curve, the appropriate parameters of the model are selected. After this, the performance, advantages and disadvantages of the model were intuitively compared by the values of accuracy, precision, recall and f-score. Finally, the selected better and more specific models are used in practical applications, that is, to predict the results based on the data of the main channel at a certain moment, and to generate win rate change images after the game to resume the game.

7.2 Future Works

First of all, there are still some over-fitting problems for some models even if the optimal parameters are selected, which may be limited by the size and number of data sets used in this project. Because it takes time to capture the data of the game, this experiment The number of data sets cannot be increased in a short time. In the future, the API may be used to capture more data to train the model. Second, whether it is the KI colonel that has been widely used in the International League of Legends game or the result prediction model made by this project, it is based on the prediction of the economic situation of both parties during the game. However, in many cases, the outcome of the game depends not only on the economic situation, but also on the player's game operation level. Under the premise of the same lineup and economic conditions, players and teams with different operating levels may achieve different results. So in the future, I consider adding some values and characteristics that can measure the player's operation level to the prediction model, such as combining the player's game history and the winning rate of the selected hero, to make a prediction,
which makes the construction of the prediction model more reasonable and the result more precise.
Third, in the course of the game, the real-time recommendation can be made in combination with the prediction of the game result at the current time point and the situation of each feature value. This needs to improve the calculation speed of the prediction model, extract the feature values and their weights, and make recommendations based on the complex weapons, equipment, skill cooling time, etc. in the game. Due to the epidemic, the items of this semester are all in my personal Is completed on the laptop, so better hardware conditions cannot be obtained. So after the hardware conditions can be met, I am interested in embarking on this research.
Bibliography

https://www.kaggle.com/datasnaek/league-of-legends


INDEPENDENT STUDY CONTRACT PROJECTS

Note: Enrolment is subject to approval by the course convener

SECTION A (Students and Supervisors)

UnitID: u6502494
Surname: Cong
First Names: Jingwen
Project Supervisor (may be external): Penny Kyburz

Course Code, Title and Units: COMP755 Individual Computing Project (12 Units)
Commencing Semester: S1, S2 YEAR: 2020 Two-semester project (12u courses only)

Project Title: Prediction of game score and results based on data of League of Legends

Learning Objectives:
The project will develop the following skills:
1. Collecting and choosing a suitable database, cleaning and data processing skills.
2. Catching data through API.
3. Multiple machine learning and deep learning algorithms and prediction model.
4. Understand and learn data science technologies which are widely used in game area.

Project Description:
With League of Legends (LOL) becoming one of the Asian Games E-sports performance competitions, E-Competition is becoming more prominent and visible. Research related to LOL is also increasing.

Due to my personal interest, my research will aim to predict the result and final score of the LOL game based on the real players and game data. The project will use the data set provided in Kaggle (https://www.kaggle.com/datasource/league-of-legends) to complete the following processes:
1. Dealing with huge data set caught by API using data processing technology. Clean and merge data sets, and integrate them into a data set that is efficient and appropriate in size and content.
2. Using multiple data mining approaches to discover the potential relationships between different features and the result or the final score of the game. By querying information and asking senior players, through the algorithm between existing features, new valuable feature values are obtained.
3. Data samples are trained by using algorithm prediction models. Through the comparison of various algorithm results, the algorithm with the best training result is selected, or several algorithms are combined to obtain a better prediction model and result. Alternative algorithms include machine learning, BP neural network model, GRNN neural network model etc.
4. Capture the real-time data of the competition and predict the results, providing the competitors with recommendations and warnings.

Research School of Computer Science

Form updated Jun 2018

49
ASSESSMENT (as per the project course’s rules web page, with any differences noted below).

<table>
<thead>
<tr>
<th>Assessed project components:</th>
<th>% of mark</th>
<th>Due date</th>
<th>Evaluated by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report: style: Research Report (e.g. research report, software description,...)</td>
<td>45</td>
<td></td>
<td>(examiner)</td>
</tr>
<tr>
<td>Artefact: kind: Software / Data (e.g. software, user interface, robot,...)</td>
<td>45</td>
<td></td>
<td>(supervisor)</td>
</tr>
<tr>
<td>Presentation:</td>
<td>(10)</td>
<td></td>
<td>(course convenor)</td>
</tr>
</tbody>
</table>

MEETING DATES (IF KNOWN):

STUDENT DECLARATION: I agree to fulfil the above defined contract:

Jingwen Cong 08/03/2020

Signature  Date

SECTION B (Supervisor):

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project. I nominate the following examiner, and have obtained their consent to review the report (via signature below or attached email)

Signature  05/04/2020

Examiner:

Name:  Signature  5/4/2020

(Nominated examiners may be subject to change on request by the supervisor or course convenor)

REQUIRED DEPARTMENT RESOURCES:

SECTION C (Course convenor approval)

Signature  Date

Research School of Computer Science  Form updated Jun 2018