Project 1

Classification of unlabeled LOL match records with “Win/Loss”

Fengjian Mao

I Introduction

i. Detailed description of problem

In this project, I have access to about 3 Million match records of solo gamers as training set. Each record comprises of all publicly available game statistics of a match played by some gamer. In sum, the fields include: Game ID, Creation Time (in Epoch format) , Game Duration (in seconds) , Season ID , Winner (1 = team1, 2 = team2) , first Baron, dragon, tower, blood, inhibitor and Rift Herald (1 = team1, 2 = team2, 0 = none) , the number of tower, inhibitor, Baron, dragon and Rift Herald kills each team has. What I need to accomplish is to create one or more classifiers that take as inputs from any fields from above (except “winner) such match record, and labels this record as a "1" or a "2".

ii. Methodology

In this project, I used pandas and numpy to preprocess the initial dataset, which included removing unnecessary features and separating the features and labels. Then, I created 4 classifiers, which are SVM, MLP, DesicionTree and KNN. I used 3 million match records to train these 4 classifiers and then used 2 milllion match records to test the performance of these 4 classifiers.

II Algorithm

i. SVM and its parameters kernel ,gamma and C

SVM is the abbreviation of support vector machine. SVM is a kind of generalized linear classifier, which classifies data according to supervised learning. Its decision boundary is the maximum margin hyperplane for solving learning samples. SVM uses the hinge loss function to calculate the empirical risk and adds a regularization term to the solving system to optimize the structural risk. SVM can classify nonlinearly by kernel method, which is one of the common kernel learning methods.

In this classifier, I chose rbf as kernel, auto for gamma and 0.9 for C. rbf kernel is also

called Gaussian kernel whose corresponding mapping function maps sample spaces into infinite dimension spaces. Gamma determines the distribution of data mapped to new feature spaces. The larger gamma is, fewer support vectors, smaller gamma values, and more support vectors. The number of support vectors affects training and prediction speed. “auto” is default value for gamma. C is penalty coefficient which can adjust weight of interval size and classification accuracy during optimization process. Finally, 0.9 is chosen for C value, since the prediction accuracy is highest when C equals 0.9.

ii. MLP and its parameters hidden_layer_sizes and max_iter

MLP is the abbreviation of Multilayer Perceptron. MLP is a feedforward artificial neural network model which maps multiple data sets into single output data sets.

In this classifier, I chose 10 for hidden_layer_sizes and 200 for max_iter. “hidden_layer_sizes” means the number of hidden layers. “max_iter” means the maximal number of iteration.

iii. DT and its parameters max_depth, criterion, splitter and min_samples_leaf

DT is the abbreviation of decision tree. DT is a prediction model, which represents a mapping relationship between object attributes and object values.

In this classifier, I chose 20 for max_depth, “entropy” as criterion, “best” as splitter and 10 for min_samples_leaf. “max_depth” means maximum depth of tree. Generally there is no need to set max_depth when sample features are less. Howeve, if sample features are too large or too characteristic, we can set an upper limit, usually 10 ~ 100. “criterion” is feature selection method. “splitter” is feature partition point selection method. “min_samples_leaf” means minimum number of samples required for leaf nodes. If this threshold is not reached, all leaf nodes of the same parent node are pruned, which is a parameter that prevents overfitting.

iv. KNN and it parameters n_neighbors, weights, algorithm and leaf_size

KNN is the abbreviation of K nearest neighbor. The KNN algorithm is given a

training dataset, for new input instances, finding the nearest instance of this instance in training data sets, most of which belong to classes, classifying this input instance into this class.

In this classifier, I chose 8 for n_neighbors, “uniform” as weights, “auto” as algorithm and 30 for leaf_size. “n_neighbors” means the number of the neighbors considered for each classification. “uniform” as “weights” means the weight for each node is the uniform. “auto” as “algorithm” can help choose the best algorithm mode from “ball_tree”, ”kd_tree” and “brute”. “leaf_size” means the number of the leaves of the tree.

III Requirements

package	python	scikit-learn	numpy	pandas
version	3.8.3	0.23.1	1.18.5	1.0.5

IV Results

i. Table containing all results

classifier	SVM	MLP	DT	KNN
time (s)	5.0±1.0	7.0±1.0	0.05±0.02	3.0±1.0
accuracy	0.9714±0.0010	0.9710±0.0010	0.9675±0.0010	0.9675±0.0010

ii. Screen capture

```
Running time: 4.740268707275391 seconds
SVM_accuracy: 0.9714368988633052

Running time: 6.875165939331055 seconds
MLP_accuracy: 0.9710968619450112

Running time: 0.045876502990722656 seconds
DT_accuracy: 0.9675993393568445

Running time: 2.743694305419922 seconds
KNN_accuracy: 0.967550762654231
```

V Comparison and discussion

i. Compare and discuss the accuracy of these 4 classifiers

According to the result table, we can find that both SVM and MLP classifiers can reach the higher accuracy 0.971, while DT and KNN classifiers just reach the accuracy 0.967. SVM classifier is good at solving problems of high dimension and nonlinearity. Also, SVM classifier has better generalization performance and can avoid neural network sturcture selection and local minima problem. Therefore, SVM classifier can reach high accuracy. As for MLP classifier, its paraller processing capability and learning ability are strong and MLP classifier is not susceptible to noise. Hence, MLP classifier has high classification accuracy. KNN classifier is likely to produce false errors for imbalanced sample classification, so its classification accurary is smaller than SVM and MLP. For imbalanced data, information gain tends to select those features with more samples, so DT classifier is easy to overfit.

ii. Compare and discuss the time complexity of these 4 classifiers

According to the result table, we can find that the DT classifier has the fastest classification speed, which is about 0.05 seconds. SVM, KNN and MLP classifiers take several seconds to finish the prediction, which are much slower than DT classifiers. DT classifier does not require any domain knowledge and parameter assumptions. It can easily convert the information of problem into classification rule. Therefore, DT classifier can handle large amounts of data within a short period of time. SVM classifier is sensitive to missing data and has large memory comsumption, which means it may take more time to finish the prediction. KNN classifier also has large memory comsumption and too much calculation. MLP classifier has much more complex sturcture, so it takes relatively more time to do the prediction for MLP classifier.

iii. Summary

Through this project, I train 4 classifiers which can be used to classify the winner according to the combat data. I develop a better understanding of these 4 classifiers. Also, I gain more experience of how to take advantage of these 4 classifiers and how to adjust the parameters of different classifiers to make them perform better. In this project, I just train SVM, MLP, DT and KNN classifiers. If more time is given, I will

try to train integrated classifiers, such as bagging, boosting, stacking and random-forest algorithm. In this project, the combat data are gained after the match. If more time is given and real-time combat data is given, I will try to train classifiers which can predict the winner according to the real-time combat data.