The following is a brief introduction to the algorithms we used.

1. AdaBoost

AdaBoost, which is called Adaptive Boosting, is a machine learning algorithm based on the boosting idea. AdaBoost is an iterative algorithm. The core idea is to use different learning algorithms for the same training set, train different weak classifiers, and then combine these weak classifiers to construct a final strong classifier.

Most Boosting methods change the distribution of data by changing the weight of the training data so that it can learn a series of different weak classifiers. In AdaBoost, there are two weights. The first is that each sample in the training set has a weight. For each sample that is misclassified by the weak classifiers in the last round of training, the weight is increased and the weight of the correct classification sample is reduced. Thus, in the next round of training, the misclassified sample is highly attended. In the second, each weak classifier has a weight. For a weak classifier with a relatively small classification error rate, the weight is increased, and the weight of the weak classifier which has a large classification error rate is reduced. Thus, the accuracy of the strong classifier in improved when the algorithm is finally integrated.

2. XGBoost

XGBoost is a gradient boosting decision tree. Its full name is Extreme Gradient Boosting, which is an extension of Gradient Boosting. The Boosting classifier is an ensemble learning model, whose basic idea is to combine hundreds of tree models that have lower classification accuracy into a model with high accuracy. The model is continuously iterated and generates a new tree for each iteration. Determining how to generate a reasonable tree at each step is the core of the Boosting classifier. The Gradient Boosting algorithm uses the idea of gradient descent in generating each tree. Then based on all the trees generated in the previous step, it moves in the direction of minimizing the given objective function. Under reasonable parameter settings, a certain number of trees need to be generated to achieve the expected accuracy. When the data set is large and complex, the Gradient Boosting algorithm has a huge amount of computation. XGBoost is an implementation of Gradient Boosting that automatically uses multithreading of the CPU for parallel operations and it could improve accuracy by improving the algorithm.

XGBoost's base learners are regression trees. Its loss function uses second-order Taylor expansion. It has high accuracy, and is not easy to overfitting, and is scalable. It can process high-dimensional sparse features distributedly. Therefore, Xgboost is 10 times faster than similar algorithms under the same circumstances.

3. SmoteBagging

SmoteBagging is an ensemble learning algorithm that uses voting strategies. As you can see from the name, the SmoteBagging algorithm is a method that uses the Smote and Bagging methods. Smote is a method that could artificially synthesize new samples, while Bagging samples the training set in a way that sampling with replacement to construct different training sets for each base classifier, and it usually adopts a simple voting method for decision output. When creating each base classifier for voting, the Nn majority sample will be sampled first, and then the same number of Nn minority samples will be constructed. The minority samples are obtained by sampling with replacement and smoting, and the proportion is determined according to the percentage b%, where b% is a multiple of 10% between 10% and 100%. That is, in each base classifier, a few samples are sampled by sampling with replacement whose ratio is Nn*b%, while the proportion of the artificially synthesized samples using Smote is Nn*(1-b%).

In each iteration, SmoteBagging can choose the method for multiplying the number of samples of majority class. In this process, the minority samples with an insufficient number are generated by smote algorithm, and selecting different numbers of majority samples in each iteration improves the difference of base classifiers. SmoteBagging is an algorithm that uses the idea of oversampling, but it does not simply use one of the methods of Smote or Bagging. It not only reduces the overfitting that the Bagging method may produce, but also reduces the negative impact of the sample of the artificial synthesis in the Smote method. In addition, since b% is a multiple of 10% between 10% and 100%, the diversity of base classifiers is guaranteed. Diversity is very important in the improvement of classification accuracy and generalization of the model.

4.EasyEnsemble

EasyEnsemble is a common algorithm that uses

http://dx.doi.org/10.21037/atm-20-1475

undersampling to deal with the imbalance problem. Unlike direct undersampling, it undersamples multiple balanced data sets and uses these data sets to train multiple classifiers. Finally, these classifiers are combined by some strategies.

EasyEnsemble can be described as follows. Suppose that the minority of the training data set is P, the majority is N, and |N| >> |P| is satisfied. The data sets of the majority are divided into N sub-data sets of N1, N2, N3...NT, and satisfy |Ni| =|P|. For each data set Ni, we combine it with P as a training set to train a classifier Hi, in which AdaBoost is used to train the classifier Hi. Finally, T classifiers are obtained and combined to form the final model.

The idea behind EasyEnsemble is very simple. In order to avoid data imbalance, the algorithm samples T-balanced sub-data sets. Each of them includes all minority and partial majority of the initial samples. In addition, a simple average method, rather than a voting method, is used in constructing the final classifier so that we can obtain information from all samples. Besides, EasyEnsemble is an ensemble algorithm based on AdaBoost, and because the AdaBoost algorithm is also an ensemble algorithm, the EasyEnsesmble algorithm is also called ensemble of ensemble algorithm.

5.Extreme Gradient Boosting forest(XGBF)

The EasyEnsesmble algorithm does not generate new data, so it learns quickly. The information provided by each sample is completely obtained by using it. However, since there are too few samples of majority class in each subset, the subclassifier cannot obtain more majority information. In addition, due to the objective distribution of data and the objective conditions of data collection, different types of samples tend to have similar values on certain features. Because these samples are located in overlapping areas of the feature space, they are called overlapping samples, and the problems caused by them are also called class overlapping problems, which may result in poor classification. The EasyEnsesmble algorithm is not ideal for this problem.

AdaBoost and XGBoost are currently better algorithms for processing general data. They are not targeted on large-scale unbalanced data sets, so it is very likely that their performance will be poor.

SmoteBagging consumes a lot of time due to its excessive smote operation and may affect the true distribution of data.

Therefore, in view of the shortcomings of the above algorithms, our study proposes an oversampling ensemble algorithm named Extreme Gradient Boosting Forest (XGBF). The algorithm combines the strong points of undersampling, oversampling and ensemble. At the same time, it also merges with XGBoost because it has the advantage of flexibility, high precision, short training time, and prevention of over-fitting.

XGBF, as an ensemble learning algorithm, is composed with several XGBoost classifiers. XGBoost itself is a boosting ensemble model using decision trees as its base classifier. The specific structure of the XGBF model is shown in Figure S1.

Since XGBoost has a fast convergence rate, the operation speed of XGBF can be guaranteed. The training data entered into each XGBoost classifier are composed with some undersampled majority data and oversampled minority data. In the XGBF algorithm, the oversampling operations are performed on the minority class, which includes duplicating and smote. The minority samples are strengthened by these two methods, so the learning model can get more information from the minority samples. The undersampling operation is performed on the majority class so that the distribution of each kind of sample is balanced. Finally, each weak subclassifier is enhanced by ensemble methods to achieve better classificion results.

Algorithm 1 XGBF Algorithm

1. i=0
2. Duplicate all minority samples in P t times to get duplicate dataset DP
3. For each sample x in DP, repeat *somte* operation n times to get dataset P'
4. While i<= tb repeat
   4.1 i=i+1
   4.2 Sampling m * |P| samples into N_i from N without replacement
   4.3 Using N_i and P' train an XGBoost classifier H_i
5. Output a weighted average of the results of tb classifier H

The XGBF algorithm combines the advantages of the oversampling, undersampling and ensemble methods so that the classifier can fully learn the characteristics of the samples and produce better results. The pseudo code for the XGBF algorithm is shown in algorithm 1. In the algorithm, P represents the minority class dataset, which is the patients' set; N represents the majority class dataset, which is the non-patient's set; and tb represents the number of XGBoost classifiers. |P| and |N| mean the cardinality of the set P and N, respectively

http://dx.doi.org/10.21037/atm-20-1475

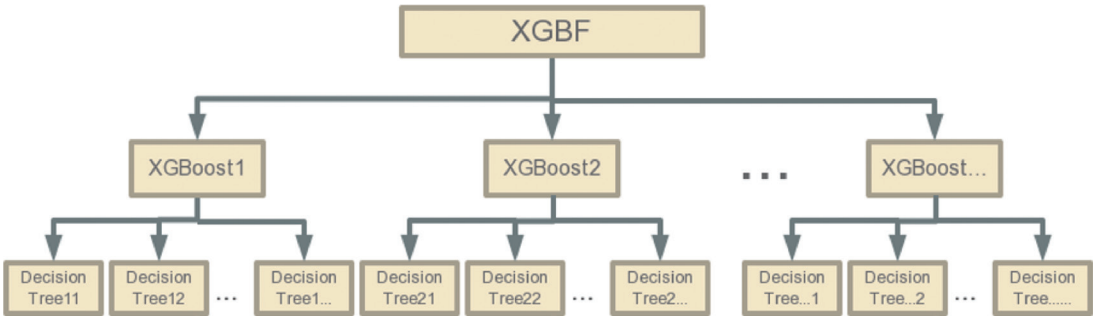

Figure S1 The specific structure of the XGBF model.

http://dx.doi.org/10.21037/atm-20-1475