PROJECT 1 REPORT

LIN YUWEI
20446054

1. Problem Description

In the training dataset, there are 3320 samples and 57 variables. Given training data and corresponding binary labels, this project need to train a classifier and predict class labels of the test dataset. Many classification models have been taken route in data analysis and most of them can always produce a satisfactory prediction if training by high-quality training dataset. Among the multiple choices of classifiers, random forest is one of the most popular and effective models. This project will build a random forest classifier to trace the pattern of the training data and make a prediction of test data. In this report, section 1 is the introduction of the project. Section 2 introduces the algorithm principle of random forest. And at last, section 3 focus on the experiment and the final result.

2. Random Forest

Random forest is one of the ensemble methods. Ensemble method is an algorithm that constructs a set of weak classifiers. These base classifiers independently make decisions and vote for the prediction. Though the classifiers may have poor performance, the ensemble classifier comes out with a lower generalization error. Based on the independent assumption between classifiers, the sum of error rate follows the accumulative binomial probability distribution. Suppose there are 25 independent base classifiers. Each classifier has error rate $\varepsilon = 0.35$, probability that the ensemble classifier makes a wrong prediction reduces to 0.06.

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

Random forest ensembles multiple decision trees and make decisions by vote. The trees are randomly constructed by bagging methods and sub-feature selection, which help relief the overfitting problem of decision tree methods.

To illustrate random forest methods mathematically, consider a random forest combined by n randomized decision trees $h(x, \theta_k), k = 1,2, \dots, n$, where x is the input dataset, $\{\theta_k\}, k = 1,2, \dots, n$ are i.i.d. random variables. The algorithm is shown below:

Step 1:Input data $L = \{(x_i, y_i), i = 1,2, \dots, n\}, y_i \in \{1,2, \dots, K\}, m$ is the number of variables to be chosen at each tree $(m \ll r), B$ is the number of bootstrap samples.

Step 2: For $b = 1,2, \dots, B$:

- Draw a bootstrap sample L^{*b} from the learning dataset L.
- From L^{*b}, use random input variables to grow a tree classifier T^{*b}. That is, randomly select a subset m of the r input variable and use the only m selected variables to determine the best split at each node (by information

gain or Gini index). To reduce bias, the tree is grown to be a maximum depth without pruning.

- The tree T^{*b} generates an associated random vector θ_b, which is independent of the previous $\theta_1, \theta_2, \ldots, \theta_{b-1}$.
- Using θ_b and an input vector *x*, a classifier $h(x, \theta_b)$ having a single vote for the class of *x* is defined.

Step 3:The B randomized tree-structured classifiers $\{h(x, \theta_b)\}$ are collectively called a random forest. And the observation x is assigned to the majority vote-getting class as determined by the random forest.

Step 4: there are two tuning parameters for a random forest: the number m of variables randomly chosen as a subset at each tree and the number B of bootstrap samples. Adjust these two parameters to get a more satisfactory results.

The generalization error for a random forest having B trees is defined as

$$PE_B = P_{X,Y}\{m_B(X,Y) < 0\}$$

where

$$m_B(X,Y) = \frac{1}{B}\sum_{b=1}^{B} I_{[h(x,\theta_B)=Y]} - \max_{k \neq y} \{\frac{1}{B}\sum_{b=1}^{B} I_{[h(x,\theta_B)=k]}\}$$

is the classification margin for the ensemble. $m_B(X,Y) > 0$ means that the ensemble model votes for the correct classification.

Since random forest starts in the same way as bagging, which samples with replacement. Suppose there are N samples and we resample N times, then a sample has a probability of $(1 - \frac{1}{\mathrm{N}})^{\mathrm{N}}$ of not being selected after the end of bagging.

$$\lim_{\mathrm{N}\to\infty}(1 - \frac{1}{\mathrm{N}})^{\mathrm{N}} = \lim_{\mathrm{N}\to\infty}[(1 - \frac{1}{\mathrm{N}})^{-\mathrm{N}}]^{-1} = \frac{1}{\lim_{\mathrm{N}\to\infty}(1 - \frac{1}{\mathrm{N}})^{-\mathrm{N}}} = \frac{1}{\mathrm{e}} \approx 0.368,$$

which means the training data will contain approximately 63.2% of the instances and 36.8% of the instances will be Out-of-Bag observations.

3. Experiment Result

The experiment is conducted using Python 3.6.0 and package sklearn.ensemble.

First, using default setting of RandomForestClassifier to run the training dataset, I get an OOB (Out-of-Bag) score **0.925155** (use Out-of-Bag observations as the test data to evaluate accuracy of the model).

Second, since the default setting of the tree number is 10, which come out a result that can be promoted. I try to adjust the tree number parameter from 100 to 400 to get a most appropriate result. The OOB score is **0.953105** when the tree number becomes 210 (the best parameter by GridSearch).

Third, use the optimized model to predict the test data and save the predicted label. The tuning parameter includes 210 bootstrap samples (trees) and 'sqrt' of variable numbers $(\sqrt{57} \approx 8)$ as max feature number.