Report of Binary Classification

Jerry Yin

July 9 2019

1 Introduction

This report talks about the classifiers I constructed to estimate whether the children is eligible for the program in the next period. Therefore, it is a binary classification. The input is the questionnaire of this period, and the output is the eligibility to the program of the next period (Y1-R1, R1-R2, etc.). The person is eligible if the risk factor is greater or equal to 5, otherwise, the person is classified as ineligible.

2 Key Metrics

I use the classification report in python as the metric of each model. There are several metrics in the classification report:

		Actual	
		Positive	Negative
Predict	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Table 1: Confusion Matrix

Precision measures in all the samples that are predicted as positive, how many are they are actually positive:

$$Precision = \frac{TP}{TP + FP}$$

Recall Rate measures in all the samples that are actually positive, how many of them are detected as positive

$$Recall = \frac{TP}{TP + FN}$$

F1 Score is the harmonic mean of the precision and recall rate:

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Macro Average is simply the average of a specific score of positive instances and negative instances.

Weighted Average is the weighted (based on number of instances) average of a specific score of positive instances and negative instances.

3 Classification Based on Single Classifier

In this section, the following classifiers are used. Each classifier is trained separately and their performances on the test data is recorded. Here 0 stands for ineligible (positive) and 1 stands for eligible (negative). As initially, the data is imbalanced (number of positive instances is four times the negative instances), we set the weight of each negative instances four times as the positive ones to get better trained results.

- K Nearest Neighbor
- Gaussian Naive Bayesian
- Support Vector Machine
- Support Vector Machine (with ν)
- Decision Tree
- Random Forest (using Gini Loss)
- Random Forest (using Entropy Loss)
- Linear Regression Classifier
- Multi-layer Perceptron

The classification report is shown as following:

Classification Report of KNN				
	Precision	Recall	F1 Score	Support
0	0.94	0.82	0.88	1587
1	0.85	0.95	0.89	1632
Accuracy			0.88	3219
Macro Avg.	0.89	0.88	0.88	3219
Weighted Avg.	0.89	0.88	0.88	3219

The result of SVC and ν-SVC are the same. I include them in one table:

Classification Report of SVC				
	Precision	Recall	F1 Score	Support
0	1.00	1.00	1.00	1587
1	1.00	1.00	1.00	1632
Accuracy			1.00	3219
Macro Avg.	1.00	1.00	1.00	3219
Weighted Avg.	1.00	1.00	1.00	3219

Classification Report of Bayesian				
	Precision	Recall	F1 Score	Support
0	0.65	0.65	0.65	1587
1	0.66	0.65	0.65	1632
Accuracy			0.65	3219
Macro Avg.	0.65	0.65	0.65	3219
Weighted Avg.	0.65	0.65	0.65	3219

Classification Report of Decision Tree				
	Precision	Recall	F1 Score	Support
0	1.00	0.78	0.87	1587
1	0.87	1.00	0.90	1632
Accuracy			0.89	3219
Macro Avg.	0.91	0.89	0.89	3219
Weighted Avg.	0.91	0.89	0.89	3219

The two random forest also provide us with the same result:

Classification Report of Random Forest				
	Precision	Recall	F1 Score	Support
0	0.86	0.89	0.87	1587
1	0.89	0.86	0.87	1632
Accuracy			0.87	3219
Macro Avg.	0.87	0.87	0.87	3219
Weighted Avg.	0.87	0.87	0.87	3219

For the linear model, as the classification report is ill defined, there is no classification report for the model.

Classification Report of MLP				
	Precision	Recall	F1 Score	Support
0	0.61	0.74	0.67	1587
1	0.68	0.54	0.60	1632
Accuracy			0.64	3219
Macro Avg.	0.65	0.64	0.64	3219
Weighted Avg.	0.65	0.64	0.64	3219

From the reports, we can observe that some classifier provide us with better result than others. Notice that the support vector machine provides us with report with 100 percent accuracy. Though it seems good, as the kernel is "rbf", which can fit any functions to any dimension. The calculation of SVM is time consuming (we will see later).

4 Classification using Bagging Technique

Bagging technique is a technique that trains several different classifiers or same classifiers with different parameters. When facing test data, those separately trained classifier will predict independently and vote to decide the correct result. Only when the majority of the classifiers make mistakes, the bagging technique will fail to predict the correct result.

First we pick the classifiers that are suitable as our base classifiers of bagging: Bayesian, Decision Tree, Random Forest, SVC, nu-SVC, Linear Model, and MLP.

There are also two voting scheme: hard and soft. Hard means that the final prediction is simply decided by the majority; Soft means that the final prediction is decided by the importance of the classifier.

Classification Report of Bagging 1				
Note: Result is formed by the form of "hard"/"soft"				
	Precision	Recall	F1 Score	Support
0	0.89/0.99	1.00/0.80	0.94/0.88	1615
1	1.00/0.83	0.88/0.99	0.94/0.90	1604
Accuracy			0.94/0.89	3219
Macro Avg.	0.95/0.91	0.94/0.89	0.94/0.89	3219
Weighted Avg.	0.95/0.91	0.94/0.89	0.94/0.89	3219

We can observe that the "hard bagging" is good at detecting 0 and precisely predict 1 (that is, all the 0 s are detected, and all the reported 1 s are trust-worthy), while the "soft bagging" is the inverse. However, none of them reaches the accuracy of 95%. Also, the classifier takes approximately 600 seconds to train (basically because of the SVCs).

Then we drop the classifiers that do not perform well and the SVCs. That is, we use Bayesian, Decision Tree, and Random Forest as base classifiers and repeated the bagging:

Classification Report of Bagging 2				
Note: Result is formed by the form of "hard"/"soft"				
	Precision	Recall	F1 Score	Support
0	0.85/0.99	0.95/0.85	0.90/0.91	1530
1	0.95/0.88	0.85/0.99	0.90/0.93	1689
Accuracy			0.90/0.92	3219
Macro Avg.	0.90/0.93	0.90/0.92	0.90/0.92	3219
Weighted Avg.	0.90/0.93	0.90/0.92	0.90/0.92	3219

Though in some metrics, the score decreases, the model still provides with a high accuracy. Moreover, the training time of the model is significantly reduced. However, the model still has the drawback that the each model (hard or soft) is not both good at detecting and predicting 0 and 1. The following section provides the solution.

5 Classification using Boosting Technique

As we know that there are weakness of the hard bagging model and soft bagging model. To overcome their drawbacks, we introduce the boosting technique (Adaboost). Boosting technique is first trained on one "weak" classifier. For those data that are predicted incorrectly by the "weak" classifier, boosting technique increases the weight of those instances. Then based on those instances, boosting trains a "stronger" classifier. The more iteration boosting technique is performed, the stronger the classifier is.

Here, we use the "soft" bagging 2 classifier as the "weak" base classifier, and train the boosting classifier using iteration n equal to 2,5,10. The time of training is recorded as reference.

Classification Report of Boosting					
		Precision	Recall	F1 Score	Support
0	$n = 2$	0.99	0.87	0.93	1593
	$n = 5$	1.00	0.91	0.95	1625
	$n = 10$	0.98	0.94	0.96	1598
1	$n = 2$	0.89	1.00	0.94	1626
	$n = 5$	0.92	1.00	0.96	1594
	$n = 10$	0.94	0.99	0.96	1621
Accuracy			$n = 2$	0.93	3219
			$n = 5$	0.95	3219
			$n = 10$	0.96	3219
Time (s)			$n = 2$	0.3162	
			$n = 5$	0.8397	
			$n = 10$	1.9388	

From the table, we can see that the training time is significantly shorter. The model based on "soft" bagging keeps the advantage of predicting 0 and detecting 1, while improve the performance of its weaknesses: detecting 0 and predicting 1 to the score of 0.94. If more iteration is performed, the result will be better.

6 My Hand Written Bagging

Beside the packages of sklearn, I hand wrote the hard and soft bagging algorithms based on all 9 classifiers. The performance of my code is better than the result of sklearn:

Model	Accuracy
Theoretical Accuracy	99.34%
Hard Bagging	99.02%
Soft Bagging	99.58%

For those people who only took the