Combine Nearest Neighbor Distance, Gaussians, and PCA Models Using Bagging Ensemble

Yaosheng Xu

2020 Fall

Abstract

Now we have three models: Nearest Neighbor Distance, Gaussians, and PCA, to calculate P-values. However, some of them are suitable for certain dataset, and some are suitable for others. To best utilize each models, I propose an ensemble method – bagging, to combine all the methods together. The goal is to make up each model's weakness. For example, we can use Gaussians Model as an outlier detector for Nearest Neighbor Distance Model in Wine Quality dataset, etc.

Introduction

The bagging method is a traditional ensemble method that uses multiple "bags" to average the performance of each models. The intuition behind this ensemble method is to average the weights on each models, so a suitable model may have a higher weight, and unsuitable models may affect little overall. There should be a sweet spot that best utilize all the models.

Detailed Approach

We need to specify three hyper-parameters: 1) *number of bags*, 2) *each model's ratio*, and 3) *number of boots*. Every bag is a model, and different bags can use the same model. A model's ratio specifies the percentage of all the bags this model occupies. So more bags you have, more close it will follow the ratio you give. Number of boots is from a strategy called "bootstrap", which create a random subset of data by sampling. Number of boots is the number of data you sample from the training data (the sampled data can be repeated). Below is how I set these hyper-parameters:

Number of bags: 10
Each model's ratio: based on dataset
Number of boots: number of training data

Restricted from the computational power, I set the *number of bags* to be 10, and *number of boots* to be the number of training data; otherwise it will be too time-consuming. I set *the ratio of each model* by first evaluating each model's performance on the given dataset. Higher performance will get higher weights.

When training, the bagging method creates multiple bags, each is a model chosen according to the model's ratio. It trains each bags individually, and stores the trained model.

When predicting, each bag will give a N x C matrix indicating the P-values, where N is the number of test data, and C is the number of classes. Bagging Method calculate the average of these P-values:

$$\text{Pvalue} = \frac{\sum_{i=0}^{n} P_i}{n}$$

where *n* is the number of bags, and P_i is the N x C P-Value matrix of the i^{th} bag.

If we want to predict a class, we can simply pick the class who has the highest P-value:

$$\hat{y} = \ argmax_k(PValue_k)$$

Performance

1) Accuracy (Train 80%, Test: 20%)

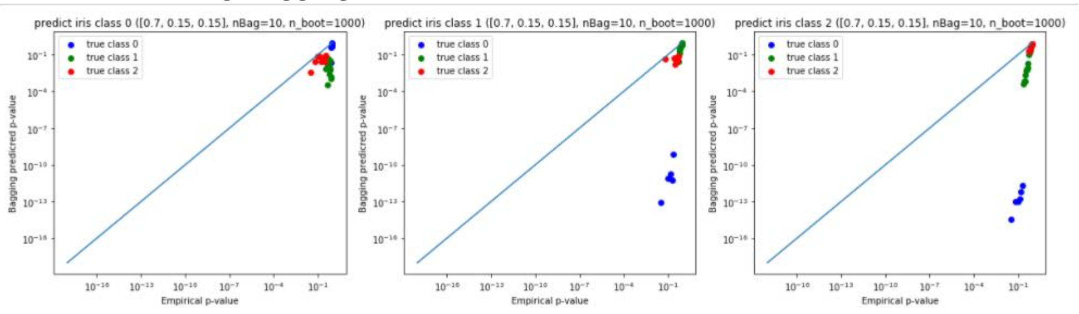
Classifier/Dataset	Iris	Wine	Wine quality
Nearest Neighbor Distance Model	0.9	0.75	0.23
Gaussian Model	0.9	0.91	0.45
PCA Model	0.9	0.86	Error
Bagging ensemble	0.93 [70%, 15%, 15%]	0.97 [0%, 50%, 50%]	0.43 [10%, 90%, 0%]

*[percentages] is the ratio for Nearest Neighbor, Gaussian, PCA.

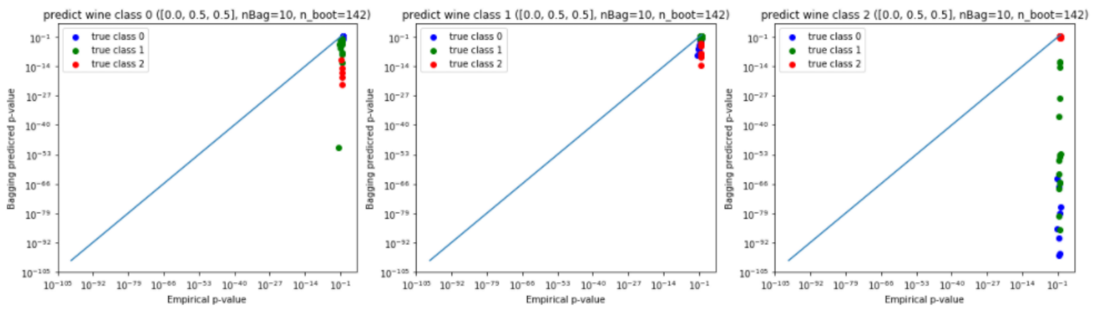
*Accuracy may flow because of random split of dataset

2) Plot

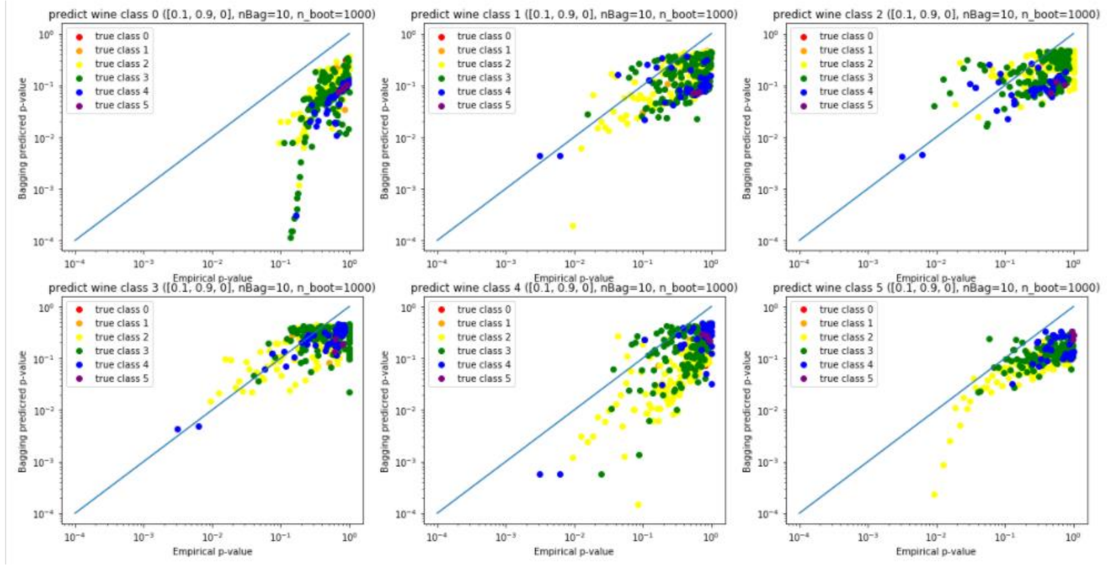
a) Iris dataset using Bagging:



b) Wine dataset using Bagging:



c) Wine quality dataset using Bagging:



Conclusion:

It seems Bagging method do improve the accuracy of some dataset, which indicate that the idea behind it is reasonable. However, for some dataset like Wine Quality, which all the models perform poorly, the Bagging method also have a bad performance.

For now, I set each model's ratio by manually evaluating each model's performance at first. In the future, we may add a linear regression method or neural network to learn the weights of each model.