Summary and discussion of: "Evidence Contrary to the Statistical View of Boosting"

Class project

Zheng Cong

May 2020

1 Summary

AdaBoost is one of the most powerful learning method. Friedman, Hastie and Tibshirani in 2000 established a statistical point of view for the AdaBoost algorithm. However, there are still many unanswered questions about AdaBoost algorithm. One of the most famous is its relative immunity to overfitting. So this paper mainly focus on illustrating the distance between statistical understanding and reality. Although AdaBoost can profitably be extended to regression, this paper[1] mainly focus on the classification problems.

There are some inconsistencies between the statistical view of boosting and the real situation observed in the experiment. Hastie claimed that overfitting are more easily caused by using trees of bigger size. However, the experiment shows that stumps are more likely to overfit. And some phenomena like this will be introduced in the section 2.

2 Experiment and Result

2.1 experiment setting and result

We just consider a two-class classification problem. The data used in the experiment comes from the model:

$$\mathrm{P}(Y=1|x) = q + (1-2q)\mathrm{I}\left[\sum_{j=1}^{J} x^{(j)} > J/2\right]$$

where X is distributed iid uniform on the d-dimensional unit cube $[0,1]^d$. q is the Bayes error and $J \leq d$ is the number of effective dimensions, n is the sample size. And the value of n, d, J, q depends on the specific experiment.

2.1.1 Stumps or more complex base classifier

It is often believed that using more complex base classifier may be easier to lead to overfitting. However, the result of the experiment shows that using stumps can overfit often.

In the experiment, we just set Bayes error rate of $q = 0.1$, training sample size $n = 200$, active dimension $J = 5$ and total dimension as $d = 20$, and test it on a 1000 samples' held

out dateset. This simulation compares the resistance of overfitting between AdaBoost using stumps and AdaBoost using 8-nodes trees.

The result is shown in Figure 1. The result shows that using more complex base classifier(8-node trees) are less likely to become overfitting.

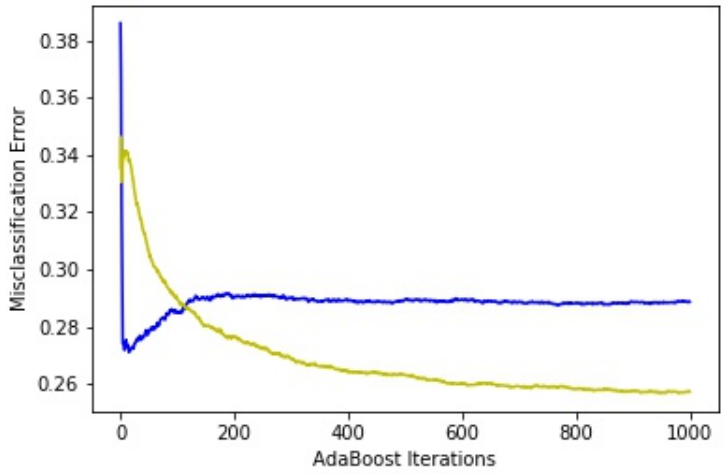
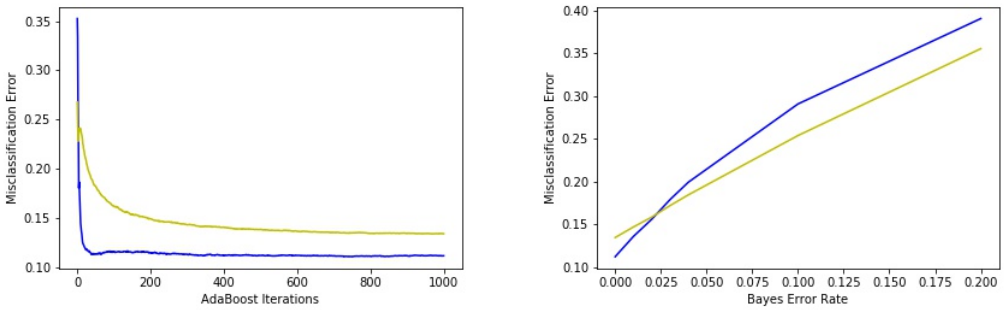


Figure 1: Comparison of AdaBoost with Stumps (Blue) and 8-Node Trees (yellow),averaging 100 times' results

2.1.2 Should Smaller Trees Be Used When the Bayes Error is Larger?

When the signal noise ratio (SNR) is small, it seems that simpler model will be preferred. Because more complex model are more likely to model the noise part, it may loss generalization ability, thus cause overfitting phenomenon. However, in the previous experiment, we find larger trees perform better when the Bayes Error Rate is high.

If we set $q = 0$, stumps has smaller classification error than 8-node trees. The result is shown in Figure 2. We could see that when the Bayes Error Rate is low, stump performs better than 8-node tree.



(a) Comparison of AdaBoost with Stumps (Blue) and 8-Node Trees (yellow),averaging 100 times' results

(b) Simulation under different Bayes Error Rate

Figure 2: Different Bayes Error Rate

2.1.3 Should LogitBoost Be Used Instead of AdaBoost for Noisy Data?

Instead of minimizing the exponential loss, LogitBoost chooses to minimize the negative binomial log likelihood. So, LogitBoost was seen as a more robust algorithm than AdaBoost, especially in noisy settings. The paper uses the experiment to compare AdaBoost and LogitBoost, where Bayes error rate $q = 0.1$.

The result is shown in Figure 3. The result tells us LogitBoost is more likely to overfit than AdaBoost in large noisy setting.

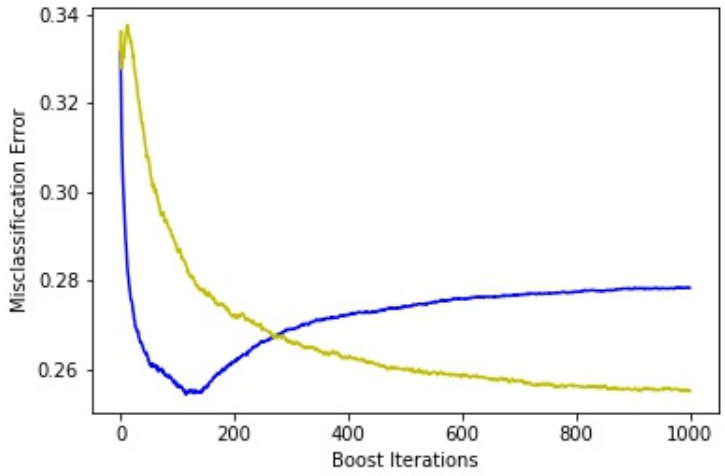


Figure 3: Comparison of AdaBoost (Blue) and LogitBoost (yellow) with 8-Node ,averaging 100 times' results

2.1.4 Should Early Stopping Be Used to Prevent Overfitting?

In machine learning, early stopping is seen as a form of regularization used to avoid overfitting when we train a learner with iterative method. And some people claim that early stopping is still applicable for the boosting algorithm. When our model complexity can be compared with true signal complexity, we should stop it early to prevent it to fit the noisy part.

In order to check whether early stopping still work, the paper just set $J = 0, q = 0.2$ in the experiment, which means there is no signal in the data. If early stopping can still work, the generalization error will increase with the increase of the iteration. However, the result of the experiment shows that self averaging property makes AdaBoost resistant to overfitting. Thus, early stopping is not needed.

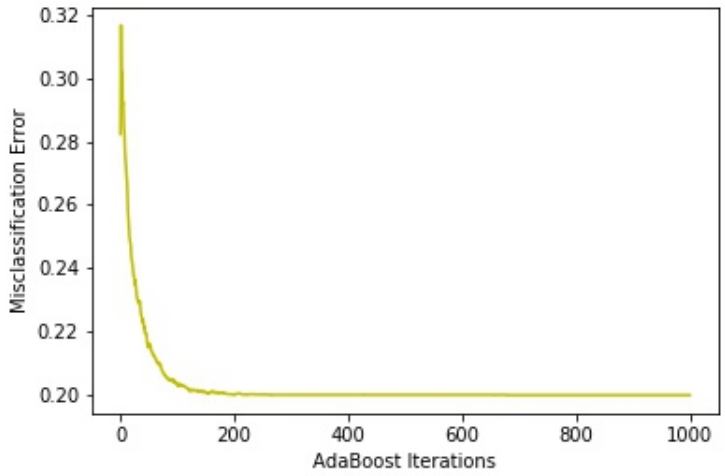


Figure 4: AdaBoost on the pure noise

2.1.5 Should Regularization Be Based on the Loss Function?

It is often considered that regularization should be based on the loss function. For example, some researchers think that early stopping should be used when the loss value becomes increased in the cross-validation dateset. For the AdaBoost algorithm, whether the exponential loss can represent the degree of overfitting.

The experiment setting is same with the 2.2.1. The exponential loss on the test dataset keeps rising. So the author claims that if we just use exponential loss to determine whether the model is overfitting, we should stop after just one iteration. However, Figure 1 tells us after 1000 iterations, the model could do best.

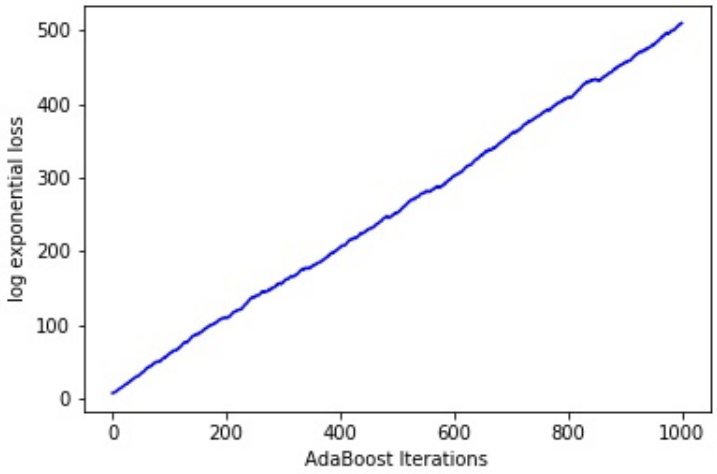


Figure 5: The Log of the Exponential Loss for AdaBoost on test dataset

2.1.6 Should the Collection of Basis Functions Be Restricted to Prevent Overfitting?

Restricting the class of trees was seen as a method to prevent overfitting. For example, if we allow the model to use all 8-node trees, the model may be more likely to overfit. Too much flexibility may cause overfitting.

In order to check whether too much flexibility will lead overfitting, the author compares the 8-node trees and 8-Node Trees restricted to have at Least 15 Observations in the leaf (terminal). Other setting is the same with section 2.1.1. We find that flexibility leads to lower misclassification error.

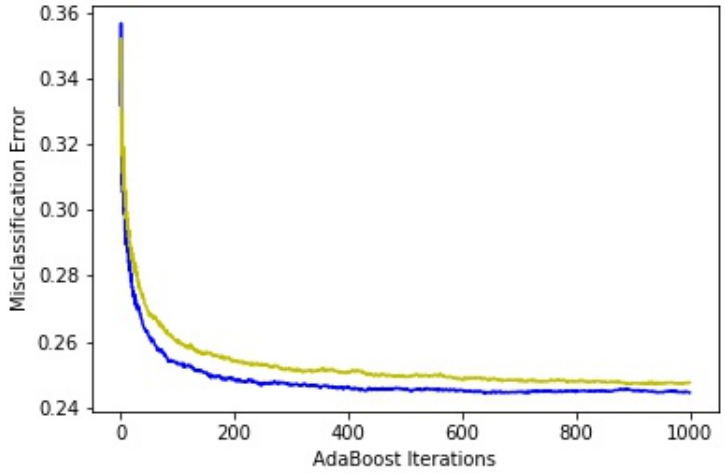


Figure 6: Comparison between 8-node trees(blue) and 8-Node Trees restricted to have at Least 15 Observations(yellow) in each leaf

2.1.7 Should Shrinkage Be Used to Prevent Overfitting?

In the statistical view of boosting, shrinkage is seen as an important regularization to resist overfitting. Shrinkage in AdaBoost means to replace α_m by $v * \alpha_m$, where v is a small number. Friedman thinks that shrinkage could improve the performance of the model and the smaller the shrinkage value, the better.

So, under the setting of section 2.1.1, the author replace α_m by $0.1 * \alpha_m$, where $v = 0.1$. From the result of the experiment, we find that shrinkage don't play an important role, instead it causes the model's overfitting.

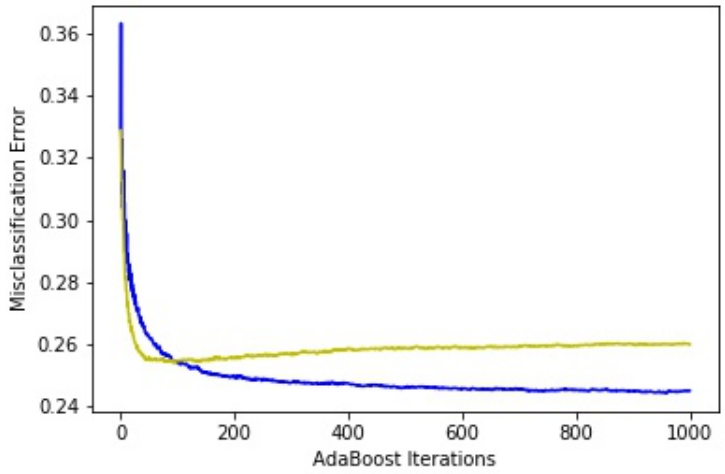
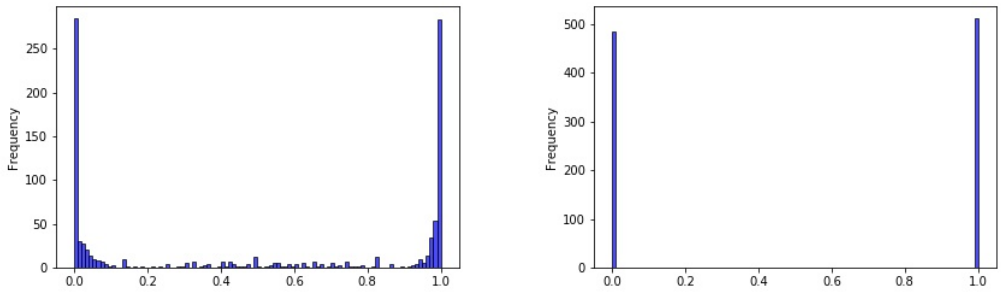


Figure 7: Comparison between 8-node trees(blue) and 8-Node Trees with shrinkage value 0.1(yellow)

2.1.8 Is Boosting Estimating Probabilities?

When we minimize the exponential loss $\sum_{i=1}^{n} e^{-y_i F_m(x_i)}$, we could derive the probability of $P(y=1|x) = \frac{\exp^{F_m}}{\exp^{F_m}+\exp^{-F_m}}$ and $P(y=0|x) = \frac{\exp^{-F_m}}{\exp^{F_m}+\exp^{-F_m}}$. However, the probabilities are often overfit and always overconfident.

The experiment setting is the same with section 2.1.1. If we set Bayes Error Rate as 0.1, the true probabilities are either 0.1 or 0.9. Thus we could compare it with the experiment result to see whether the probability is overconfident or not. The result is shown in the Figure 8. The result of iteration 10 seems more reasonable than iteration 1000. We find that most probabilities are greater than 0.99 or less than 0.01, which is not consistent with the true probability. The result shows the result of AdaBoost is overconfident. In fact, section 2.1.5 has indicated this kind of result, the score function F_m always keeps increasing.



(a) Probability estimation from AdaBoost when m=10 iteration

(b) Probability estimation from AdaBoost when m=1000 iteration

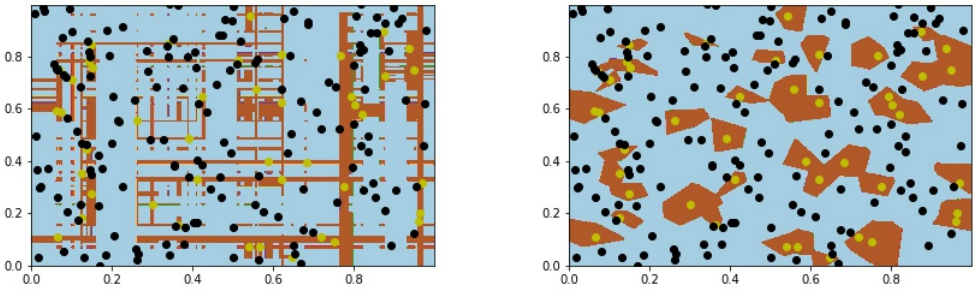
Figure 8: Probability estimation from AdaBoost

2.1.9 Is Boosting Similar to the One Nearest Neighbor Classifier?

AdaBoost can achieve zero classification error in the training dataset, which is similar to the nearest neighbor classifier. So, some researchers try to connect AdaBoost with nearest neighbor classifier. They find they are very similar under the 1 dimension situation.

The equivalence between AdaBoost and nearest neighbor classifier will not hold in the 2 dimension situation. That means equivalent behavior on the training data does not imply similar performance for the whole population. The experiment just repeats the section 2.1.4, and set $d = 2$.

The result is shown in Figure 9. We can clearly see the difference between AdaBoost and nearest neighbor classifier.



(a) AdaBoost with 8-node tree

(b) Nearest Neighbor Classifier

Figure 9: The population level difference between AdaBoost and Nearest Neighbor Classifier

2.1.10 Is Boosting Consistent?

For any estimator, we always need to check whether or not it is consistent. For the AdaBoost algorithm, the author suggest us to compare the final accuracy with the parameter Bayes Error Rate. If the estimator is consistent, the final misclassification error should be equal to Bayes Error Rate. And when the sample size is large enough, they should be arbitrarily close to each other.

In the experiment, the author sets $J = 1, d = 5, n = 5000, q = 0.1$ and the test sample size is 1000. The result is shown in Figure 10. We could find after 1000 iterations, the misclassification is very close to 0.1. Thus, AdaBoost is almost consistent.

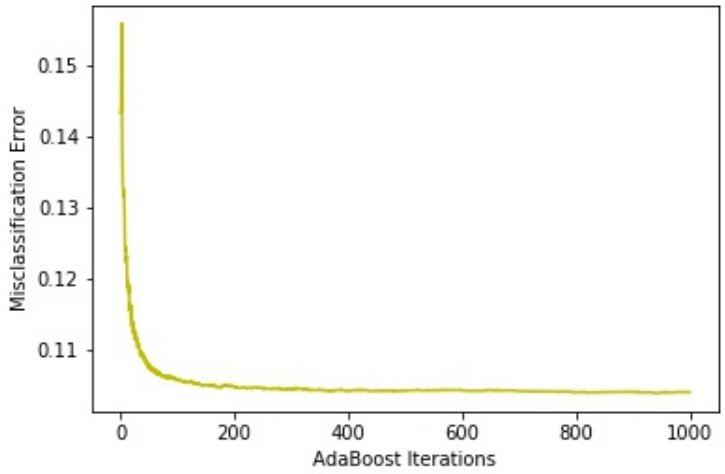


Figure 10: Performance of AdaBoost with Bayes Error Rate of 0.1

3 discussion

The author introduces some interesting phenomena around the overfitting problem, which is not consistent with the traditional understanding. One of the most interesting phenomenon is Boosting's immunity to overfitting and it seems that Boosting algorithm prefers large base classifier, which can be seen from section 2.1.1, 2.1.2, 2.1.4. Compared to stumps, 8-node base classifier is less likely to be overfitting.

Another important discovery mentioned in the paper is that the probability given by the boosting algorithm is not reasonable. The result of section 2.1.5 and 2.1.8 shows that boosting algorithm seems always overconfident and the situation becomes worse as the increase of the iteration. From this point of view, boosting algorithm is still overfitting and calibrating the conditional class probability seems more challenge than just estimating the median.

From my point of view, these two discoveries are highly correlated. For the lack of overfitting, some researchers attribute it to the stagewise nature of the boosting algorithm and provide some empirical evidence. And for the overfitting problem, it also can be seen as a result of stagewise additive operation.

References

[1] David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *Journal of Machine Learning Research*, 9(Feb):131–156, 2008.