Boosted Perceptron Classifier Performance Analysis

Mary Nathalie G. Dela Cruz
2015-09114 AI 201 PA 4 Tuesday Class

Boosting is one of the most commonly used methods in ensemble learning. It is based on the idea that combining multiple weak learners into an ensemble of classifiers enables them to perform better than any weak learner alone. Here, a weak learner is a learning algorithm that has slightly better training accuracy than random guessing. One of the most popular boosting methods is the Adaptive Boosting (AdaBoost) algorithm, which was proposed by Freund and Schapire. In each weak learning cycle of the AdaBoost algorithm, a set of weights corresponding to the training samples was adjusted such that the weights of the misclassified samples were increased while the weights of the correctly classified samples were decreased. [1,2]

To implement the AdaBoost algorithm, we trained the Perceptron Classifier as our weak learner using the Pocket Algorithm. We evaluated the performance of the Boosted Perceptron on two non-linearly separable datasets: banana and splice. The banana dataset consisted of 5300 two-dimensional vectors labeled as either banana (+1) or not banana (-1). On the other hand, the splice dataset consisted of 2991 60-dimensional vectors labeled as either intron boundaries (+1) or non-boundaries (-1). Both were binary classification problems.

The implementation was divided into two sections: (1) Perceptron Classifier Construction and, (2) AdaBoost Construction and Evaluation. In the construction of the Perceptron Classifier, we implemented and tested the Pocket Algorithm for perceptual learning. This was performed on a random, normally distributed dataset that consisted of 100 two-dimensional vectors labeled as (+1) or (-1). In the construction of the AdaBoost algorithm, we implemented the AdaBoost Algorithm with the Pocket Algorithm as the basic learner and used it for the classification of test data. This was implemented on the banana dataset and the splice dataset. To evaluate the classifier performance on the datasets, we varied the number of iterations or learners K from 10 to 1000, plotted the accuracy of the classifier on the training and the test set as a function of K, and compared the results.

Below is the plot and the table of training and test set accuracies as a function of the number of learners (K) for the banana dataset and the splice dataset. As shown, the boosted perceptron achieved high accuracy for both datasets. For 10, 20...,990, and 1000 learners, the splice test set had an average accuracy of 78.31%, reaching up to 80.76% of maximum accuracy at K=910, while the splice train set had an average accuracy of 92.02%, reaching up to 99.30% of maximum accuracy at K=930. On the other hand, the banana test set had an average accuracy of 87.25%, reaching up to 89.53% of maximum accuracy at K=810, while the banana train set had an average accuracy of 91.13%, reaching up to 95.75% of maximum accuracy at K=880.

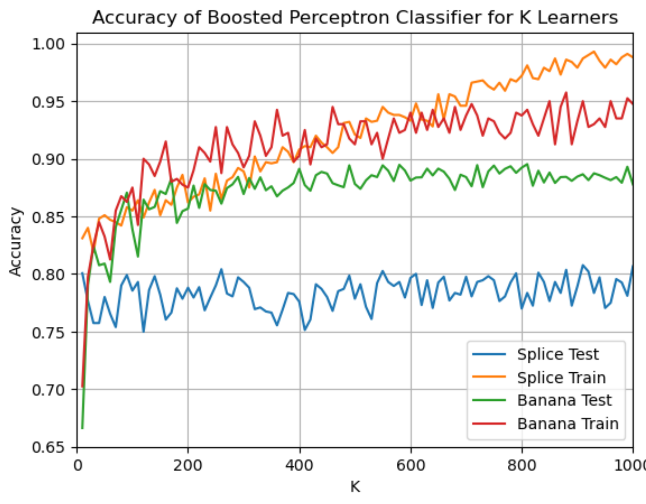
The accuracy increased for both datasets but at different rates. For the test set and train set of the banana dataset, accuracy increased rapidly in the first 100 iterations and then it balanced out. On the other hand, the accuracy increased gradually throughout the k iterations for the train set of the splice dataset but only slightly increased for the test set of the splice dataset. These findings may indicate that the banana dataset was easier to learn compared to the splice dataset. The boosted perceptron also encountered a generalization problem with the splice dataset that contained 60 features because it failed to increase its accuracy compared to the banana dataset.

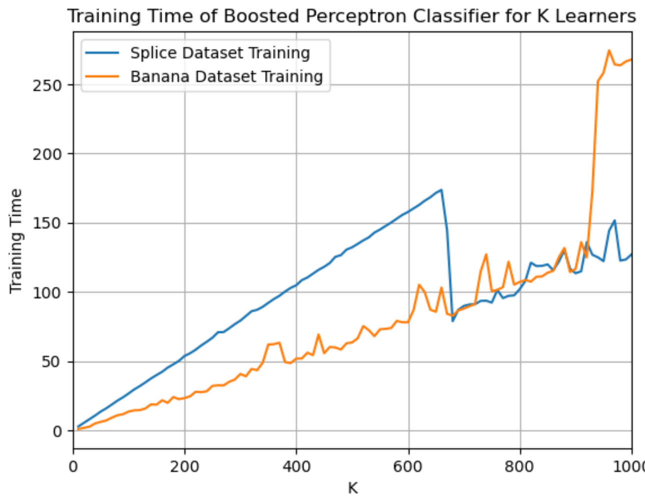
The gap between the accuracy of the classifier on the test set and the train set of the banana dataset was small and around 5%. This may imply that the classifier avoided overfitting the banana dataset. On the other hand, the gap for the splice dataset was larger and increased with k iterations. This may imply that the classifier did not avoid overfitting with the splice dataset.

Given that the training time increases with K, the optimal performance can be achieved if K is set between 310 to 400 for the banana dataset and 210 to 300 for the splice dataset. To improve the performance of the boosted perceptron on the splice dataset, feature selection techniques or dimensionality reduction techniques can be used. A learning rate can also be added to the algorithm to help the classifier converge.

In conclusion, the Boosted Perceptron Classifier can achieve high accuracy on different datasets by adjusting the number of learners or iterations. In this project, we evaluated the classifier performance on two datasets—banana and splice—by plotting its accuracy versus the number of learners or iterations K. We found that the accuracy of the classifier can reach up to 80.76% on the splice test set, 99.30% on the splice train set, 89.53% on the banana test set and 95.75% on the banana train set. We also found that the accuracy increased at different rates for both datasets where the banana dataset seemed easier to learn compared to the splice dataset which encountered a generalization problem during its testing. We found that there is a gap between the accuracy for the test set and the train set where testing for the banana dataset avoided overfitting while testing for the splice dataset had indications of overfitting. To achieve optimal performance, we recommend setting the number of iterations or learners to be between 310 to 400 for the banana dataset and to be between 210 to 300 for the splice dataset.

K Range	**Splice Dataset**		**Banana Dataset**	
	Test Ave Accuracy	**Train Ave Accuracy**	**Test Ave Accuracy**	**Train Ave Accuracy**
10-100	0.7766	0.8436	0.8097	0.8275
110-200	0.7790	0.8646	0.8573	0.8850
210-300	0.7864	0.8772	0.8724	0.9055
310-400	0.7729	0.8977	0.8775	0.9153
410-500	0.7787	0.9164	0.8814	0.9195
510-600	0.7870	0.9348	0.8855	0.9240
610-700	0.7868	0.9435	0.8850	0.9323
710-800	0.7881	0.9657	0.8881	0.9313
810-900	0.7868	0.9784	0.8841	0.9338
910-990	0.7894	0.9868	0.8847	0.9381





Reference:

[1] Ferreira, Artur J., and Mário AT Figueiredo. "Boosting algorithms: A review of methods, theory, and applications." *Ensemble machine learning: Methods and applications* (2012): 35-85.

[2] Ying, Cao, et al. "Advance and prospects of AdaBoost algorithm." *Acta Automatica Sinica* 39.6 (2013): 745-758.