Machine Learning – Lab 7
AdaBoost

Introduction

AdaBoost is a machine learning meta-algorithm that can be used in conjunction with many other types of learning algorithms to improve their performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier.

AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems, however, it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing (i.e., their error rate is smaller than 0.5 for binary classification), the final model can be proven to converge to a strong learner.

Boosting (Schapire 1989)

1. Randomly select n1 < n samples from D without replacement to obtain D1
    a. Train weak learner C1
2. Select n2 < n samples from D with half of the samples misclassified by C1 to obtain D2
    a. Train weak learner C2
3. Select all samples from D that C1 and C2 disagree on
    a. Train weak learner C3
4. Final classifier is vote of weak learners

Adaboost (Adaptive Boosting)

Given training data $(x_1, y_1),...,(x_m, y_m)$, $y_i \in \{-1, 1\}$, $x_i \in X$ is the object or instance, y_i is the classification.

1. for t = 1,..,T
    a. create distribution D_t on {1,...,m}
    b. select weak classifier with smallest error ϵt on D_t
        i. $\epsilon_t = Pr_{Dt}[h_t(x_i) != y_i]$
        ii. $h_t : X \rightarrow \{-1,+1\}$
2. output single classifier $H_{final}(x)$

At each iteration of the training process a weight is assigned to each sample in the training set equal to the current error on that sample. These weights can be used to inform the training of the weak learner, for instance, decision trees can be grown that favor splitting sets of samples with high weights.

To Do

1. Install ScikiLearn and dependencies
    a. https://scikit-learn.org/stable/install.html
    b. see attached makefile
2. Read, understand, implement and run this example
    a. https://scikit-learn.org/stable/auto_examples/ensemble/plot_adaboost_multiclass.html
    b. attached source file adaboost.py
3. Add a single decision tree classifier to the sample and compare errors to adaboost
    a. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
    b. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

Other Examples

- https://scikit-learn.org/stable/auto_examples/ensemble/plot_adaboost_multiclass.html
- https://scikit-learn.org/stable/auto_examples/ensemble/plot_adaboost_hastie_10_2.html
- https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html
- https://scikit-learn.org/stable/auto_examples/ensemble/plot_adaboost_twoclass.html
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

Bibliography

[1] http://math.mit.edu/~rothvoss/18.304.3PM/Presentations/1-Eric-Boosting304FinalRpdf.pdf
[2] http://en.wikipedia.org/wiki/AdaBoost
[3] https://www.cs.princeton.edu/~schapire/papers/explaining-adaboost.pdf
[4] http://www.nickgillian.com/wiki/pmwiki.php?n=GRT.AdaBoost