11

Classification

11.1 Introduction

(REVISED)

11.2 Boosting

Let T be a training set,

$$T = \{(x_1, y_1), \ldots, (x_m, y_m)\},$$

where $x_i \in X$ and $y_i \in \{-1, 1\}$. The labels $y_1, \ldots, y_m$ specify the class where each of the examples x_i belong; namely, we have $y_j = 1$ of x_j is a positive example and $y_j = -1$ if x_j is a negative examples. If $S = \{x_1, \ldots, x_m\}$, a classifier is regarded as a function $h : S \longrightarrow \{-1, 1\}$. Note that a classifier places an example x in its correct class if $xh(x) = 1$.

AdaBoost, short for *Ada*ptive *Boost*ing, is a machine learning algorithm, formulated by Yoav Freund and Robert Schapire. AdaBoost starts with a family of weak classifiers $\mathcal{W} = \{h_1, \ldots, h_t\}$, that is, with a collection of classifiers that have high error rates and seeks to build a strong classifier h. Namely, if $h(x) = \sum_{t=1}^{t_{max}} \alpha_t h_t(x)$ is a linear combination of the weak classifiers we seek f as $f(x) = sign(h(x))$ for $x \in S$.

The construction process is sequential. The weak classifier used at moment t is h_t. At each moment t we have a probability distribution D_t that gives greater weight to examples that were misclassified in a previous step. We have

$$\sum_{i=1}^{m} D_t(x_i) = 1.$$

In the initial step, $t = 1$, the distribution is uniform, that is,

$$D_1(x_1) = \cdots = D_1(x_m) = \frac{1}{m}.$$

Starting from the distribution D_t the distribution $D_{t+1}(x_i)$ is given by

$$D_{t+1}(x_i) = D_t(x_i)\frac{e^{-\alpha_t y_i h_t(x_i)}}{Z_t},$$

where Z_t is a normalization factor intended to ensure that $\sum_{i=1}^m D_{t+1}(x_i) = 1$. Thus, Z_t is given by

$$Z_t = \sum_{i=1}^m D_t(x_i)e^{-\alpha_t y_i h_t(x_i)}.$$

Since $e^{-\alpha_t y_i h_t(x_i)} < 1$ when $y_i = h_t(x_i)$ and $e^{-\alpha_t y_i h_t(x_i)} > 1$ when $y_i \neq h_t(x_i)$, it follows that $D_{t+1}(x_i) > D_t(x_i)$, when h_t is wrong on x_i and $D_{t+1}(x_i) < D_t(x_i)$, when h_t is correct on x_i.

Theorem 11.1. *Let D be a probability distribution on the set $S = \{x_1, \ldots, x_m\}$ and let $h : S \longrightarrow \{-1, 1\}$ be a classifier. Define*

$$Z(\alpha) = \sum_{i=1}^m D(x_i)e^{-\alpha y_i h(x_i)}.$$

The minimum value of $Z(\alpha)$ is achieved when

$$\alpha = \frac{1}{2}\ln\frac{1-\epsilon}{\epsilon},$$

where ϵ is the probability of error

$$\epsilon = \sum_{i=1}^m \{D(x_i) | y_i \neq h_i(x_i)\}.$$

Proof. We have

$$\begin{aligned}\frac{dZ}{d\alpha} &= \sum_{i=1}^m -y_i h(x_i)D(x_i)e^{-\alpha y_i h(x_i)}\\ &= -\sum\{D(x_i)e^{-\alpha}|y_i = h(x_i)\} + \sum\{D(x_i)e^{\alpha}|y_i \neq h_i(x_i)\}\\ &= -e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon,\end{aligned}$$

Thus, the value of α for which Z is minimal is given by $\alpha = \frac{1}{2}\ln\frac{1-\epsilon}{\epsilon}$.

Theorem 11.2. *For the training error of $f(x) = sign(h(x))$ we have:*

$$\frac{1}{m}\Big|\{i | h(x_i) \neq y_i\}\Big| \leq \prod_{t=1}^T Z_t.$$

Proof. Note that

$$\begin{aligned}
D_{t+1}(x_i) &= \frac{D_1(x_i)}{\prod_{i=1}^t Z_i} e^{-\sum_{t=1}^t \alpha_t y_i h_t(x_i)} \\
&= \frac{1}{m\prod_{i=1}^t Z_i} e^{-\sum_{t=1}^t \alpha_t y_i h_t(x_i)} \\
&= \frac{1}{m\prod_{i=1}^t Z_i} e^{-y_i \sum_{t=1}^t \alpha_t h_t(x_i)} \\
&= \frac{1}{m\prod_{i=1}^t Z_i} e^{-y_i h(x_i)}.
\end{aligned}$$

The last equality implies

$$e^{-y_i h(x_i)} = m D_{t+1}(x_i) \prod_{i=1}^t Z_i. \tag{11.1}$$

For the classifier $f = sign\left(\sum_{t=1}^{t_{max}} \alpha_t h_t\right)$ define

$$\chi_f(x) = \begin{cases} 1 & \text{if } f(x) \neq y, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, $\chi_f(x) = 1$ if and only if the classifier f erred on x. Thus, the error rate of f is $\epsilon_f = \frac{1}{m}\sum_{i=1}^m \chi_f(x_i)$.

If $f(x_i) \neq y_i$, then we have either $f(x_i) = 1$ (and therefore, $h(x_i) > 0$) and $y_i = -1$, or $f(x_i) = -1$ (and therefore, $h(x_i) < 0$) and $y_i = 1$. Thus, in either case we have $y_i h(x_i) \leq 0$, which implies $e^{-y_i h(x_i)} > 1$ which implies $\chi_f(x_i) \leq e^{-y_i h(x_i)}$. Consequently, taking into account Equality (11.1), we have

$$\frac{1}{m}\sum_{i=1}^m \chi_h(i) \leq \frac{1}{m}\sum_{i=1}^m e^{-y_i h(x_i)} = \sum_{i=1}^m \left(\prod_{t=1}^T Z_t\right) D_{T+1}(x_i) = \prod_{t=1}^T Z_t.$$

This allows us to conclude that $\prod_{t=1}^T Z_t$ is an upper bound of the training error.

Since $\alpha_t = \frac{1}{2}\ln\frac{1-\epsilon_t}{\epsilon_t}$ it follows that

$$\begin{aligned}
Z_t &= \sum_{i=1}^m D_t(x_i) e^{-\alpha_t y_i h_t(x_i)} \\
&= \sum_i \{D_t(x_i) e^{-\alpha_t} \mid y_i = h_t(x_i)\} + \sum_i \{D_t(x_i) e^{\alpha_t} \mid y_i \neq h_t(x_i)\} \\
&= (1-\epsilon_t)e^{-\alpha_t} + \epsilon_t e^{-\alpha_t} \\
&= 2\sqrt{\epsilon_t(1-\epsilon_t)}.
\end{aligned}$$

Algorithm 11.2.1: The Adaboost Algorithm

```
   Data: A data set $(x_1, y_1), \ldots, (x_m, y_m)$, where $x_i \in X$ and $y_i \in \{-1, 1\}$
   Result: A boosted classifier $h$
 1 initialize weights $D_1(x_i) = \frac{1}{m}$ for $1 \leq i \leq m$
 2 for $t = 1$ to $t_{max}$ do
 3     select a training set drawn from the distribution $D_t$;
 4     train $h_t$ such that $\epsilon_t = \sum_{i=1}^{m} \{D_t(x_i) \mid y_i \neq h_t(x_i)\}$ is minimal;
 5     if $\epsilon_t \geq 0.5$ then
 6         reset $D_t$ to $D_1$ and abandon $h_t$
 7     end
 8     set $\alpha_t = 0.5 \log \frac{1-\epsilon_t}{\epsilon_t}$;
 9     update $D_{t+1}(x_i) = \frac{1}{Z_t} \cdot D_t(x_i) e^{-\alpha_t y_i h_t(x_i)}$;
10 end
11 $f(x) = \sum_{t=1}^{T} \alpha_T h_t(x)$
12 return $h(x) = sign(f(x))$
```

The pseudocode of the AdaBoost is shown in Algorithm 11.2.1. The algorithm maintains a weight distribution $D_t(x_i)$ on the training instances $x_1, \ldots, x_m$ from which the data subset S_t is chosen for each classifier h_t. Initially, the distribution is uniform, so all instances have equal chances to participate in the training set. The training error ϵ_t is also weighted by the distribution, such that ϵ_t is the sum of the distribution weights of the instances misclassified by h_t. We require that this error be less than $\frac{1}{2}$ (which is the error rate of a classifier that would assign classes at random).

11.3 Bagging

Bagging is a technique invented by L. Breiman [3]. The term “bagging” is an acronym of *bootstrap aggregating.*

As before, a *learning set* consists of a data set $\mathfrak{L} = \{(x_i, y_i) \mid 1 \leq i \leq m\}$, where y_i is either the class label or a numerical label. Assume that we have an algorithm for using this learning set to form a predictor $\phi(x, \mathfrak{L})$.

Suppose that we are given a sequence of learning sets $(\mathfrak{L}_k)$, each consisting of m_k independent observations from the same underlying distribution as $\mathfrak{L}$. Our goal is to use the sequence $(\mathfrak{L}_k)$ to get a better predictor than $\phi(x, \mathfrak{L})$ by using the sequence of predictors $(\phi(x, \mathfrak{L}_k)$.

If y is numerical one could use the average of $\phi(x, \mathfrak{L}_k)$, that is $\phi_A(x) = E_{\mathfrak{L}}(\phi(x, \mathfrak{L}))$, where $E_{\mathfrak{L}}$ is expectation over $\mathfrak{L}$ and A denotes aggregation.

If $\phi(x, \mathfrak{L})$ predicts a class j, $1 \leq j \leq J$, then one method of aggregating is by voting.

Usually, we have a single learning set $\mathfrak{L}$. Still, an imitation of the process leading to ϕ_A can be done by taking *bootstrap samples* $\{\mathfrak{L}^{(B)}\}$ from $\mathfrak{L}$ and form $\{\phi(x, \mathfrak{L}^{(B)}\}$. If y is numerical take ϕ_B as

$$\phi_B(x) = afg_B \phi(x, \mathcal{L}^{(B)}).$$

The $\mathcal{L}^{(B)}$ form replicate the data set, each consisting of n cases, drawn at random, but with replacement from $\mathcal{L}$. Each (y_n, x_n) may appear repeated times or not at all in any particular $\mathcal{L}^{(B)}$.

A critical factor in whether bagging improves the accuracy is the stability of the procedure for constructing ϕ. If changes in $\mathcal{L}$ produces small changes in ϕ, then ϕ_B will be close to ϕ. Improvement will occur for *unstable* procedures, where a small change in $\mathcal{L}$ can result in a large change in φ. So, for unstable procedures, bagging works well.

MATLAB Computations

Exercises and Supplements

1.

Bibliographical Comments