模式识别 week 10 Ensemble

Problem 1. Boosting

1.1

$$E = e^{-\frac{\alpha_m}{2}} \sum_{n \in I_m} w_n^{(m)} + e^{\frac{\alpha_m}{2}} \sum_{n \in M_m} w_n^{(m)}$$

当 y_m 确定. α_m 的最优值是 $\left(\frac{A}{X} + BX\right)$ 在 $e^{\frac{\alpha_m}{2}} = \sqrt{\frac{\sum_{n \in I_m} w_n^{(m)}}{\sum_{n \in M_m} w_n^{(m)}}}$ 取得.

即 $\alpha_m = \ln \frac{\sum_{n \in I_m} w_n^{(m)}}{\sum_{n \in M_m} w_n^{(m)}}$

又 $\varepsilon_m = \frac{\sum_{n \in M_m} w_n^{(m)}}{\sum_{n=1}^{N} w_n^{(m)}} \Rightarrow \alpha_m = \ln \frac{1-\varepsilon_m}{\varepsilon_m}$

1.2

$$w_n^{(m+1)} = \exp(-t_n f_m(x_n)) = w_n^{(m)} \cdot \exp(-t_n \frac{\alpha_m}{2} y_m(x_n))$$

$$-t_n y_m(x_n) = 2 I(y_m(x_n) \neq t_n) - 1$$

$$\therefore w_n^{(m+1)} = w_n^{(m)} \exp(\alpha_m I(y_m(x_n) \neq t_n)) / \exp(\frac{\alpha_m}{2})$$

对于 $\forall n \in$ range(1, N). 有 $\exp(\frac{\alpha_m}{2})$ 为 const. 在 normalize 之后可以忽略

$$\therefore w_n^{(m+1)} = w_n^{(m)} \exp(\alpha_m I(y_m(x_n) \neq t_n))$$

Problem2

2.1.

单个 decision stump 的错误率可以写为:

$$\sum_{i=1}^{n} w_i I\{h_{a,d,j}(x_i) \neq y_i\} = \sum_{i=1}^{n} \frac{w_i\left(1 - y_i h_{a,d,j}(x_i)\right)}{2}$$

我们需要调整的参数为 a,d,j，可以先不管$\sum_{i=1}^{n} \frac{w_i}{2}$ 这个常数项。问题变为最小化 $\sum_{i=1}^{n} -w_i y_i h_{a,d,j}(x_i)$即最大化$F = \sum_{i=1}^{n} w_i y_i h_{a,d,j}(x_i)$。Decision bump 的参数 a，可以限制一定取训练数据中的某个数，因为如果将数据按照顺序排列，取$a \in [x_i, x_{i+1})$与取$a = x_i$等效。将训练数据按照第 j 维大小顺序排列后，根据 d 的两种取值，decision bump 分为 neg-pos 型和 pos-neg 型，在 neg-pos 型时，F 可以按照训练 feature 在第 j 维的大小写成如下形式:

$$F_{np} = -\sum_{x_k[j] \le a} w_k y_k + \sum_{x_k[j] > a} w_k y_k$$

$$= -\sum_{k \le j} w_k y_k + \sum_{k > j} w_k y_k$$

$$= \sum_{k=1\ldots n} w_k y_k - 2\sum_{k \le j} w_k y_k$$

当为 pos-neg 型时$F_{pn} = -F_{np} = -\sum_{k=1\ldots n} w_k y_k + 2\sum_{k \le j} w_k y_k$。所以已知 dim 之后，要找到最佳切分 index j 只要找到使$abs(F_{np})$最大的 index 即可。分析至此，算法就很简单了，找到 decision bump 参数的算法总结如下：

```
[~, p] = size(X);
max_dim_score = 0;
for dim=1:p % 对于所有feature维度循环
    [sorted_X, ind] = sort(X(:, dim)); % 按照这一维度的feature排序
    sorted_wy = y(ind) .* w(ind); % 得到[w_k * y_k] 数组
    cumsum_front = cumsum(sorted_wy); % 得到\sum_{k<=j}[w_k * y_k]数组
    whole_sum = sum(sorted_wy);
    pos_scores = whole_sum - cumsum_front * 2; % 得到F_np
    [dim_score, max_ind] = max(abs(pos_scores)); % 得到最佳划分index max_ind
    if dim_score > max_dim_score
        max_dim_score = dim_score;
        k = dim;
        a = sorted_X(max_ind); % a = x[max_ind]
        if pos_scores(max_ind) > 0 % 判断是F_np里找到最大还是F_pn里找到最大
            d = -1;
        else
            d = 1;
        end
    end
end
```

对于所有 feature 维度循环 一共 p 个循环，每个循环调用一次 sort O(n logn)，一次 element wise 乘，一次 cumsum 一次 sum，一次 abs，一次 max，均为 O(n)。O(1)做判断并赋值。总的复杂度为$O(pn\log n)$。

2.2.

update_weight.m: 训练数据的 weight 更新: 用了 Problem1 里简单的更新公式:

$$w_n^{(m+1)} = w_n^m \exp\left(-\frac{alpha_m}{2} y_n h(x_n)\right)$$

代码如下：

```
%%% Your Code Here %%%
p = ((X(:, k) <= a) - 0.5) * d; % predicted label / 2
w_update = w .* exp(-p .* y * alpha);
w_update = w_update / sum(w_update);
%%% Your code Here %%%
end
```

adaboost_error.m: 代码如下:

```
n = length(y);
legal_inds = k > 0;
k = k(legal_inds);
a = a(legal_inds);
d = d(legal_inds);
alpha = alpha(legal_inds);

ps = ((X(:, k) <= repmat(a', n, 1)) - 0.5) .* repmat(d', n, 1) * 2;
p = sign(ps * alpha);
e = (sum(p ~= y)) / n;
```

2.3.

运行 300 个 iteration 结果如下：

可以看到，随着 iteration 增加，weak learner 个数的增加，对训练集的拟合越来越好，在 test 集上的 generalization 性质还行，在收敛之后，test error 没有随着训练 iteration 持续增多有显著提升。