Homework 2 for “Machine Learning”

Instructor: Prof. Jun Zhu

October 24, 2021

Requirements:

- We recommend that you typeset your homework using appropriate software such as LaTeX. If you submit your handwritten version, please make sure it is cleanly written up and legible. The TAs will not invest undue effort to decrypt bad handwritings.
- We have programming tasks in each homework. Please submit the source code together with your homework. Please include experiment results using figures or tables in your homework, instead of asking TAs to run your code.
- You should choose one problem from 1.2 and 1.3. To relieve the burden, *no* bonus points will be given if you do both.
- Please finish your homework independently. In addition, **you should write in your homework the set of people with whom you collaborated**.
- If you have any problems, feel free to contact with me: lucheng.lc15@gmail.com.

1 Boosting: from Weak to Strong (4pt)

We re-state the AdaBoost algorithm here.

```
Algorithm 1: AdaBoost
Input: Training dataset $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$. ($y_i \in \{+1, -1\}$.) Base learner $A$.
Initialize the observation weights $D_1(i) = 1/N$, $i = 1, 2, ..., N$;
Initialize the ensembled predictive function $f_0 = 0$;
for $t = 1, 2, \ldots, T$ do
    Use $A$ to fit a classifier $h_t$ to the training data using weights $D_t(i)$;
    Compute the weighted error $\mathcal{E}_t = \sum_{i=1}^N D_t(i) I(y_i \neq h_t(x_i))$;
    Let $\beta_t \leftarrow \frac{1}{2} \ln \frac{1-\mathcal{E}_t}{\mathcal{E}_t}$ and $f_t \leftarrow f_{t-1} + \beta_t h_t$;
    Let $D_{t+1}(i) \leftarrow D_t(i) \exp(-\beta_t y_i h_t(x_i))/Z_t$, where $Z_t = \sum_{i=1}^N D_t(i) \exp(-\beta_t y_i h_t(x_i))$;
Output: $F(x) = \text{sign}(f_T(x))$ as the classifier.
```

Problem 1.1 (1pt)**.** Prove the claim in class (P26-27 in slides) that AdaBoost is equivalent to stagewise minimization of the exponential loss $\mathcal{L}(f) := \sum_{i=1}^N \exp(-y_i f(x_i))$. In other words, prove that

$$(\beta_t, h_t) = \arg \min_{\beta \geq 0, h} \sum_{i=1}^N \exp(-y_i(f_{t-1}(x_i) + \beta h(x_i))),$$

where β_t, h_t are defined in Algorithm 1.

Choose one problem from 1.2 and 1.3. *No* bonus points will be given if you do both.

Problem 1.2 (3pt)**.** We prove some nice properties of AdaBoost in this problem.

1. (2pt) A weak (but useful) learning algorithm should satisfy $\mathcal{E}_t < \frac{1}{2}$. Let $\gamma_t = \frac{1}{2} - \mathcal{E}_t$. Show that the training error at iteration T can be bounded by

$$\frac{1}{N}\sum_{i=1}^{N} I(y_i \neq F(x_i)) \leq \prod_{t=1}^{T} \sqrt{1-4\gamma_t^2}. \tag{1}$$

Remark. If there exists γ s.t. $\gamma_t \geq \gamma > 0$ $\forall t$, using Eq.(1) we can bound the training error by $(1-4\gamma^2)^{T/2}$.

Hint. First observe that the 0-1 loss is upper bounded by the exponential loss (P28 in slides), i.e.

$$I(y_i \neq F(x_i)) \leq \exp(-y_i f_T(x_i)).$$

Then prove the following identity

$$\frac{1}{N}\sum_{i=1}^{N} \exp(-y_i f_T(x_i)) = \prod_{t=1}^{T} Z_t$$

by expanding Z_T. Finally, show that

$$Z_t = \sqrt{1-4\gamma_t^2}.$$

2. (1pt) Show that the training error at margin θ satisfies

$$\frac{1}{N}\sum_{i=1}^{N} I(y_i f_T(x_i) \leq \theta) \leq \exp(\theta)\prod_{t=1}^{T} \sqrt{1-4\gamma_t^2}.$$

Remark. We mentioned in class that AdaBoost often continues to improve test error even after training error reaches zero. The calculation here leads to an explanation based on the *margin theory*, in which the training error at margin θ is related to the test error / generalization performance. Note that we simplified the problem. In margin theory, we should consider the margin for the *normalized* predictive function $\frac{1}{\sum_{i=1}^{T}|\beta_t|} f_T$, so the situation is actually more complicated. You can refer to Theorem 7.7 in *Foundations of Machine Learning* for the bound of the correct margin definition ready for use in generalization theory.

Problem 1.3 (3pt)**.** In this problem, we apply the result of Eq. 1 to bound the training error in a simple problem.

We consider a 1-dimensional dataset $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ in which $x_i \in \mathbb{R}$, $y_i \in \{+1, -1\}$. We additionally assume that x_i are distinct and $x_1 > x_2 > \ldots > x_N$. For the base learner A, we consider the *thresholding-based decision stumps* as follows, indexed by a threshold s and sign $+/-$, such that

$$h_{s,+}(x) = \begin{cases} 1 & if\ x \geq s \\ -1 & if\ x < s \end{cases}$$

and

$$h_{s,-}(x) = \begin{cases} -1 & if\ x \geq s \\ 1 & if\ x < s \end{cases}$$

And we suppose the base learner A could find the best classifier h to any weighted version of the dataset, among the all thresholding-based decision stumps.

1. (2pt) Show that for any weights $\{p_i\}$ such that $p_i \geq 0$ and $\sum_{i=1}^{N} p_i = 1$, defining $\mathcal{E}(h) = \sum_{i=1}^{N} p_i I(y_i \neq h(x_i))$, there exists a thresholding-based decision stump h such that $\mathcal{E}(h) \leq \frac{1}{2} - \gamma$, where $\gamma = \frac{1}{2N}$.

Hint. What is the difference between $\mathcal{E}(h_{x_i,+})$ and $\mathcal{E}(h_{x_{i+1},+})$? What is the relationship between $\mathcal{E}(h_{x_i,+})$ and $\mathcal{E}(h_{x_i,-})$?

2. (1pt) Based on the above answer, give an upper bound on the number of thresholding-based decision stumps (i.e. the iteration T in Algorithm 1) required to achieve zero error on a given training set. You should make your result concise by applying suitable approximation assuming N is large.

Hint. Use Eq. 1. We require the training error to be less than $\frac{1}{N}$ to achieve zero error.

2 Initialization in DNNs (2pt)

Consider a deep feed-forward network with ReLU activation. To simplify notations we do not consider bias, so the network can be written as

$$x_i^l = \sum_{j=1}^{N_{l-1}} W_{ij}^l \sigma(x_j^{l-1}), \quad 1 \le i \le N_l, 2 \le l \le D,$$

where $x_i^1 = \sum_{j=1}^{N_0} W_{ij}^1 x_j^0$ $(1 \le i \le N_1)$, $\boldsymbol{x}^0 = (x_i^0)_{i=1}^{N_0}$ denotes the input, $\boldsymbol{x}^D$ denotes the output, and $\sigma(x) := \max(x, 0)$ is the ReLU function. The weights $\boldsymbol{W} = \{W_{ij}^l\}$ are initialized with independent normal distributions:

$$p_{init}(\boldsymbol{W}) = \prod_{l,i,j} \mathcal{N}(W_{ij}^l | 0, \sigma_l^2).$$

Problem 2.1 (1pt)**.** Under a reasonable initialization scheme, the variance of the network output $\mathrm{Var}_{\boldsymbol{W}\sim p_{init}}(\boldsymbol{x}^D)$ should neither explode nor vanish, when the network depth or width grow unbounded. A sufficient condition for this is

$$\mathrm{Var}_{init}(x_i^l) = \mathrm{Var}_{init}(x_j^{l-1}) \ \ \forall l > 1, i, j. \tag{2}$$

Suppose that $\{x_i^{l-1} : 1 \le i \le N_{l-1}\}$ are i.i.d., and the distribution of x_i^{l-1} is symmetric[1][2]. Further suppose $\sigma_l = \alpha / \sqrt{N_{l-1}}$. Find α such that (2) holds. (Hint: show that $\mathbb{E}\left[\sigma(x_i^{l-1})^2\right] = \frac{1}{2}\mathrm{Var}(x_i^{l-1})$.)

Problem 2.2 (1pt)**.** Another condition the initialization scheme should satisfy is that, as the network depth or width grows unbounded, the back-propagated gradient signal,

$$\mathrm{Var}_{init}\left[\frac{\partial \mathsf{Loss}(\boldsymbol{W})}{\partial x_i^l}\right]$$

should neither explode nor vanish (so that the gradient w.r.t. weights will not explode or vanish).

Assume that $N_l = N_{l+1}$, $\{\partial \mathsf{Loss}(\boldsymbol{W})/\partial x_j^{l+1} : 1 \le j \le N_{l+1}\}$ are i.i.d., $\{\partial \mathsf{Loss}(\boldsymbol{W})/\partial x_j^{l+1}\}_{1 \le j \le N_{l+1}}$, $\{W_{ji}^{l+1}\}_{1 \le j \le N_{l+1}}$ and x_i^l are independent[3], and x_i^l follows a symmetric distribution. Show that the initialization scheme determined in the previous problem also satisfies this condition.

3 Clustering: Mixture of Multinomials (4pt)

3.1 MLE for multinomial

Problem 3.1 (1pt)**.** Derive the maximum-likelihood estimator for the parameter $\boldsymbol{\mu} = (\mu_i)_{i=1}^d$ of a multinomial distribution:

$$P(\boldsymbol{x}|\boldsymbol{\mu}) = \frac{n!}{\prod_i x_i!} \prod_i \mu_i^{x_i}, \quad i = 1, \cdots, d \tag{3}$$

where $x_i \in \mathbb{N}$, $\sum_i x_i = n$ and $0 < \mu_i < 1$, $\sum_i \mu_i = 1$.

3.2 EM for mixture of multinomials

Consider the following mixture-of-multinomials model to analyze a corpus of documents that are represented in the bag-of-words model.

Specifically, assume we have a corpus of D documents and a vocabulary of W words from which every word in the corpus is token. We are interested in counting how many times each word appears in each

[1] recall the randomness comes from initialization

[2] intuitively, these conditions can be satisfied by considering the infinite-width limit.

[3] while the independence assumption here seems implausible, you should derive the result formally assuming it holds. It can be made rigorous by assuming a different set of weight samples are used in back-propagation.

document, regardless of their positions and orderings. We denote by $T \in \mathbb{N}^{D\times W}$ the word occurrence matrix where the wth word appears T_{dw} times in the dth document.

According to the mixture-of-multinomials model, each document is generated i.i.d. as follows. We first choose for each document d a *latent* "topic" c_d (analogous to choosing for each data point a component z_n in the mixture-of-Gaussians) with

$$P(c_d = k) = \pi_k,\ k = 1, 2, \cdots, K; \tag{4}$$

And then given this "topic" $\boldsymbol{\mu}_k = (\mu_{1k}, \ldots, \mu_{Wk})$ which now simply represents a categorical distribution over the entire vocabulary, we generate the word bag of the document from the corresponding multinomial distribution[4]

$$P(d|c_d = k) = \frac{n_d!}{\prod_w T_{dw}!} \prod_w \mu_{wk}^{T_{dw}},\ \text{where } n_d = \sum_w T_{dw}. \tag{5}$$

Hence in summary

$$P(d) = \sum_{k=1}^{K} P(d|c_d = k)P(c_d = k) = \frac{n_d!}{\prod_w T_{dw}!} \sum_{k=1}^{K} \pi_k \prod_w \mu_{wk}^{T_{dw}}. \tag{6}$$

Problem 3.2 (1pt)**.** Given the corpus T, derive the EM algorithm to learn the parameters $\{\boldsymbol{\pi}, \boldsymbol{\mu}\}$ of this mixture model.

Problem 3.3 (2pt)**.** Implement the EM algorithm on the 20 Newsgroups dataset[5].

You can set the number of topics K to different values, such as $10, 20, 50, 100$, and show the 10 most frequent words in each topic for each case. Observe and analyze the results under different K.

[4] Make sure you understand the difference between a categorical distribution and a multinomial distribution. You may think about a Bernoulli distribution and a binomial distribution for reference.

[5] http://ml.cs.tsinghua.edu.cn/~shuyu/sml/20news.zip