Homework 2 for “Machine Learning”

Instructor: Prof. Jun Zhu
Course ID. #80245013

April 7, 2018

1 Boosting: From Weak to Strong

Boosting takes a *weak learning algorithm* - any learning algorithm that gives a classifier that is slightly better than random - and transforms it into a *strong classifier*, which does much better than random. In this problem, we show, using a binary classification problem on $\mathbb{R}^1$ as example, that we can get a strong classifier which achieves *zero* error on training dataset by boosting simple *thresholding-based decision stumps.*

Here we first list some definitions and existing results on boosting for reference. Given a training dataset $\{(x^{(i)}, y^{(i)})\}_{i=1}^m$, we call a vector $p = \{p^{(i)}\}_{i=1}^m$ a distribution on the examples if $p^{(i)} \geq 0,\ \forall i$ and $\sum_{i=1}^m p^{(i)} = 1$.

Definition 1. *A weak learner with margin $\gamma > 0$ is that, if for any distribution p on the m training examples there exists one weak classifier ψ_j such that*

$$\sum_{i=1}^{m} p^{(i)} 1\{y^{(i)} \neq \phi_j(x^{(i)})\} \leq \frac{1}{2} - \gamma$$

Definition 2. *The thresholding-based decision stumps are functions on $x \in \mathbb{R}^1$, indexed by a threshold s and sign $+/$, such that*

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

and

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

That is, $\varphi_{s,+}(x) = -\varphi_{s,-}(x)$.

Theorem 1. *(Convergence of Boosting) If in each iteration, the boosting procedure (similar to the algorithm shown in the lecture notes) can generate a weak classifier with margin γ, then*

$$J_t \leq \sqrt{1 - 4\gamma^2} J_{t-1},$$

where J_t denotes as the error rate on training dataset after the t-th iteration. Obviously, if $J_t < 1/m$, we have zero training error.

We consider the following binary classification problem on $\mathbb{R}^1$ with training dataset $\{(x^{(i)}, y^{(i)})\}_{i=1}^m$, where $x^{(i)} \in \mathbb{R}^1, y \in \{1, -1\}$. We additionally assume that $x^{(i)}$ are distinct and

$$x^{(1)} > x^{(2)} > ... > x^{(m)}.$$

We would like to guarantee that for any distribution p on the training set, there is some $\gamma > 0$ and a threshold s such that,

$$\sum_{i=1}^m p_i 1\left\{y^{(i)} \neq \phi_{s,+}(x^{(i)})\right\} \leq \frac{1}{2} - \gamma$$

or

$$\sum_{i=1}^m p_i 1\left\{y^{(i)} \neq \phi_{s,-}(x^{(i)})\right\} \leq \frac{1}{2} - \gamma$$

That is, in each iteration we get a weak classifier with margin γ. Hence zero training error can be achieved by boosting.

1. Show that for each threshold s, there is some $m_0(s) \in \{0, 1, ..., m\}$ such that

$$\sum_{i=1}^m p_i 1\{\phi_{s,+}(x^{(i)}) \neq y^{(i)}\} = \frac{1}{2} - \frac{1}{2}\left(\sum_{i=1}^{m_0(s)} y^{(i)} p_i - \sum_{i=m_0(s)+1}^{m} y^{(i)} p_i\right)$$

and

$$\sum_{i=1}^m p_i 1\{\phi_{s,-}(x^{(i)}) \neq y^{(i)}\} = \frac{1}{2} - \frac{1}{2}\left(\sum_{i=m_0(s)+1}^{m} y^{(i)} p_i - \sum_{i=1}^{m_0(s)} y^{(i)} p_i\right)$$

Treat sums over empty sets of indices as zero, so that $\sum\limits_{i=1}^{0} a_i = 0$ for any a_i, and similarly $\sum\limits_{i=m+1}^{m} a_i = 0$.

Hint: Note that

$$1\{y = -1\} = \frac{1-y}{2}, \quad 1\{y = 1\} = \frac{1+y}{2}$$

2. Define, for each $m_0 \in \{0, 1, ..., m\}$

$$f(m_0) = \sum_{i=1}^{m_0} y^{(i)} p_i - \sum_{i=m_0+1}^{m} y^{(i)} p_i$$

Show that $\gamma = \frac{1}{2m}$ satisfies that

$$\max_{m_0} |f(m_0)| \geq 2\gamma$$

Hint: Show that

$$|f(m_0) - f(m_0+1)| \geq \frac{2}{m}$$

3. Based on the above answer, how large margin γ can thresholded decision stumps guarantee on any training set $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$? (i.e. how "good" a weak classifier can we get in each boosting iteration?) Give an upper bound on the number of thresholded decision stumps required to achieve zero error on a given training set.

2 Deep Neural Networks

To make neural networks work well in practice is not easy in general, since there are too many hyper-parameters to tune such as the choices of the number of hidden layers, the activation function, the learning rate and so on. Besides some general guidelines (some standard techniques which are useful in most cases such as dropout, data augmentation), experiences are of great importance.

Though a beginner may often be confused with them, luckily, there are some softwares available on the Internet to help you build up a good sense on tuning neural networks.

In this problem, you need to train the neural networks with different choices of hyper-parameters from the following link - A Neural Networks Playground - and answer the following questions:

1. Identify the best configuration you find for different problems and datasets. Here you only need to list you configuration for the last dataset of the classification problem.

2. List your findings that how the learning rate, the activation function, the number of hidden layers and the regularization influence the performance and convergence rate.

3. List your "principle" which will guide you in the future tuning and explain your intuition.

3 Clustering: Mixture of Multinomials

3.1 MLE for multinomial

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

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

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 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

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

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[1]

$$P(d|c_d = k) = \frac{n_d!}{\prod_w T_{dw}!} \prod_w \mu_{wk}^{T_{dw}},\ \text{where } n_d = \sum_w T_{dw}. \tag{3}$$

Hence in summary

$$P(d) = \sum_{k=1}^{K} P(d|c_d = k)P(c_d = k) = \frac{n_d!}{\prod_w T_{dw}!} \sum_{k=1}^{K} \pi_k \prod_w \mu_{wk}^{T_{dw}}. \tag{4}$$

Given the corpus T, please design and implement an EM algorithm to learn the parameters $\{\boldsymbol{\pi}, \boldsymbol{\mu}\}$ of this mixture model and test it on the "20newsgroup" dataset[2].

Set the number of topics K to be $5, 10, 20, 30$ respectively and show the most-frequent words in each topic for each case. Observe the result and try to find the "best" K value for this dataset and explain why.

[1] 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.

[2] See http://qwone.com/ jason/20Newsgroups for details.