CS480/680: Introduction to Machine Learning

Homework 3

Due: 11:59 pm, March 9, 2021, submit on Crowdmark (yet to be set up, stay tuned).

Include your name and student number!

Submit your writeup in pdf and all source code in a zip file (with proper documentation). Write a script for each programming exercise so that the TAs can easily run and verify your results. Make sure your code runs!
[Text in square brackets are hints that can be ignored.]

Exercise 1: Kernels (5 pts)

For the following questions you might find it useful to recall the definition of a matrix being positive semidefinite (PSD). A matrix $M \in \mathbb{R}^{d\times d}$ is PSD if and only if $x^T M x \geq 0$ for all vectors $x \in \mathbb{R}^d$. Also, Taylor series are a thing.

1. (2 pt) For $x, y \in \mathbb{R}$, consider the kernel function $k(x,y) = \exp\left(-\alpha\,(x-y)^2\right)$. What is the corresponding feature map $\phi(\cdot)$ such that $\phi(x)^T\phi(y) = k(x,y)$? If you were using this kernel for an SVM model, would you prefer to solve the primal or dual representation? Why?

   Ans:

2. (1 pt) Consider the function $\frac{1}{1-xy}$, where $x, y \in (-1, 1)$. Is this function a valid kernel? If so, write out the corresponding feature map $\phi(\cdot)$, if not, explain why.

   Ans:

3. (1 pt) Consider the function $\log(1 + xy)$, where $0 < x, y \in \mathbb{R}$. Is this function a valid kernel? If so, write out the corresponding feature map $\phi(\cdot)$, if not, explain why.

   Ans:

4. (1 pt) Consider the function $\cos(x + y)$, where $x, y \in \mathbb{R}$. Is this function a valid kernel? If so, write out the corresponding feature map $\phi(\cdot)$, if not, explain why.

   Ans:

Exercise 2: Decision Trees (5 pts)

Consider the dataset given below. X_1 and X_2 are the features and Y is the class label:

X_1	X_2	Y
0.5	2.5	−
0.5	5.5	+
1.5	4.5	−
2.5	1.5	+
2.5	4.5	−
4.5	1.5	−
4.5	4.5	+

Recall that in decision trees, we pick an attribute X_j and partition the dataset into two subsets according to a threshold t: $D_l = \{i : X_{i,j} < t\}$ and $D_r = \{i : X_{i,j} \geq t\}$. On each subset D_s, we count the frequency of positives p^s_+ and the frequency of negatives p^s_-, for $s \in \{l, r\}$. Then, we evaluate our split using the Gini index

$$\mathrm{GI}(t,j) = \sum_{s\in\{l,r\}} \sum_{y\in\{+,-\}} p^s_y(1 - p^s_y). \tag{1}$$

(Empty sum is treated as 0.)

1. (1 pt) Consider attribute X_1, what is the best t?

   Ans:

2. (1 pt) Consider attribute X_2, what is the best t?

   Ans:

3. (1 pt) With the best t for X_1 and X_2, which attribute will you choose, X_1 or X_2? Explain your answer.

   Ans:

4. (1 pt) Draw a Decision Tree (need not be the best one) that has 100% accuracy on the given dataset.

   Ans:

5. (1 pt) Will your decision tree correctly classify the following test point: $X_1 = 0.5, X_2 = 7.5, Y = +$. Explain your answer.

   Ans:

Exercise 3: Adaboost (5 pts)

Recall the update rules of Adaboost:

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^n w_j^t}, \quad i = 1, \ldots, n \tag{2}$$

$$\epsilon_t = \epsilon_t(h_t) = \sum_{i=1}^n p_i^t \cdot [\![h_t(\mathbf{x}_i) \neq y_i]\!] \tag{3}$$

$$\beta_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t} \tag{4}$$

$$w_i^{t+1} = w_i^t \exp(-y_i \beta_t h_t(\mathbf{x}_i)), \quad i = 1, \ldots, n. \tag{5}$$

Here we use i and t to index the training examples and iterations, respectively. $[\![A]\!] = 1$ if the event A holds and 0 otherwise. Unlike in class, here $y_i \in \{\pm 1\}$ and **we also assume** $h_t(\mathbf{x}_i) \in \{\pm 1\}$. In this exercise we offer additional insights about Adaboost.

1. (1 pt) Prove by induction that the updates defined above indeed are equivalent to what we learned in class:

$$\tilde{\beta}_t = \frac{\epsilon_t}{1 - \epsilon_t} \tag{6}$$

$$\tilde{w}_i^{t+1} = \tilde{w}_i^t \tilde{\beta}_t^{1 - |\tilde{h}_t(\mathbf{x}_i) - \tilde{y}_i|}, \tag{7}$$

   where in class we used $\tilde{y}_i = \frac{y_i + 1}{2}, \tilde{h}(\mathbf{x}_i) = \frac{h(\mathbf{x}_i) + 1}{2} \in \{0, 1\}$.

   [Hint: Show that p_i^t remains the same under the two seemingly different updates.]

   Ans:

2. (1 pt) Recall in the logistic regression lecture we made the linear assumption on the log odds ratio:

$$\log \frac{p(y = 1 | \mathbf{X} = \mathbf{x})}{p(y = -1 | \mathbf{X} = \mathbf{x})} \approx \mathbf{w}^\top \mathbf{x} + b. \tag{8}$$

   Adaboost in effect tries to approximate the log odds ratio using *additive* functions:

$$\log \frac{p(y = 1 | \mathbf{X} = \mathbf{x})}{p(y = -1 | \mathbf{X} = \mathbf{x})} \approx \sum_{t=1}^T \beta_t h_t(\mathbf{x}), \tag{9}$$

   where each $h_t(\mathbf{x})$ is a weak classifier. Indeed, fixing $\mathbf{X} = \mathbf{x}$, prove that the minimizer of the following exponential loss

$$\min_{H \in \mathbb{R}} \quad \mathbb{E}[\exp(-yH) | \mathbf{X} = \mathbf{x}] \tag{10}$$

is (proportional to) the log odds ratio. Here the expectation is wrt the conditional distribution $p(y|\mathbf{X} = \mathbf{x})$.

[Any function can be approximated arbitrarily well by additive functions but clearly not by linear functions, thus the power of Adaboost.]

Ans:

3. (1 pt) Suppose h_t is a weak classifier whose error $\epsilon_t > 1/2$, i.e. worse than random guessing! In this case it makes sense to flip h_t to $\bar{h}_t(\mathbf{x}) = -h_t(\mathbf{x})$. Compute the error $\epsilon_t(\bar{h}_t)$ and the resulting $\bar{\beta}_t$. Do we get the same update for w in (5)? Explain.

   Ans:

4. (1 pt) Adaboost is a greedy algorithm where we find the weak classifiers sequentially. At iteration t, the classifiers $h_s, s < t$ are already found along with their coefficients β_s. Suppose h_t is given by some oracle, to find the optimal coefficient β_t, we solve an empirical approximation of the exponential loss (10):

$$\min_{\beta \in \mathbb{R}} \ \frac{1}{n} \sum_{i=1}^{n} \exp(-y_i[H_{t-1}(\mathbf{x}_i) + \beta h_t(\mathbf{x}_i)]), \tag{11}$$

   where needless to say, $H_{t-1}(\mathbf{x}) := \sum_{s=1}^{t-1} \beta_s h_s(\mathbf{x})$. While it is possible to solve (11) directly, we gain more insights by defining a distribution over the training examples $(\mathbf{x}_i, y_i), i = 1, \ldots, n$:

$$p_i^t = \frac{\exp(-y_i H_{t-1}(\mathbf{x}_i))}{\sum_{i=1}^{n} \exp(-y_i H_{t-1}(\mathbf{x}_i))}, \tag{12}$$

   so that we can rewrite (11) equivalently as:

$$\min_{\beta \in \mathbb{R}} \ \hat{\mathbb{E}}_t \exp[-y\beta h_t(\mathbf{x})], \quad (\mathbf{x}, y) \sim \mathbf{p}^t. \tag{13}$$

   (Here the hat notation is to remind you that this is an empirical expectation specified by $\mathbf{p}^t$ over our training data.) Let ϵ_t be defined as in (3). Prove that the optimal β in (13) is given in (4).

   [Hint: We remind again that both y and $h_t(\mathbf{x})$ are $\{\pm 1\}$-valued. Split training examples according to $h_t(\mathbf{x}_i) = y_i$ or not.]

   Ans:

5. (1 pt) What is the training error

$$\epsilon_{t+1}(h_t) = \sum_{i=1}^{n} p_i^{t+1} \cdot [\![h_t(\mathbf{x}_i) \neq y_i]\!] \tag{14}$$

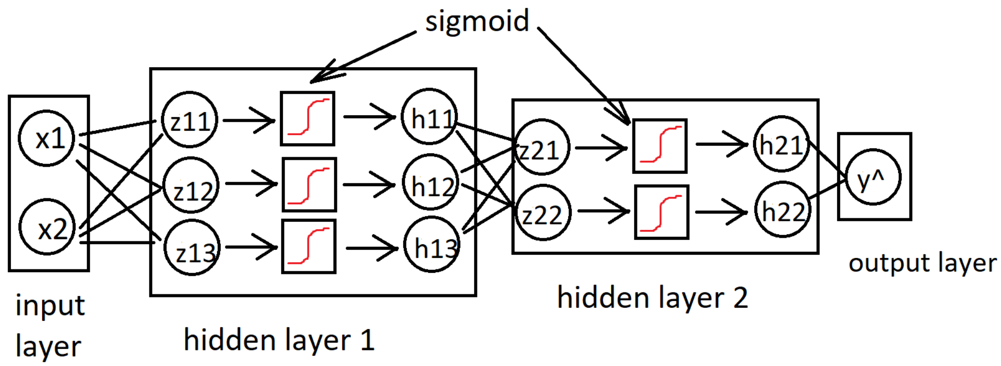
   of the weak classifier h_t on the next round $t + 1$? Justify your answer. You may assume $0 < \epsilon_t < 1$ so that all quantities are well-defined.

   [Hint: This exercise should be simple, given what you have done in Ex 2.4. Split training examples according to $h_t(\mathbf{x}_i) = y_i$ or not.]

   Ans:

Exercise 4: Backpropagation (5 pts)

Suppose we have a multilayer perceptron:



Recall that the sigmoid function is $f(x) = \frac{1}{1+e^{-x}}$.
Here

$$z_1 = Ux + c, \quad U = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad c = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad z_2 = Vh_1 + d, \quad V = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad d = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

$$\hat{y} = \langle h_2, w \rangle + b, \quad w = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad b = 0.5$$

Also suppose that the dataset

$$D = \{(x_1, y_1), (x_2, y_2)\} = \left\{ \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, 0 \right), \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}, 1 \right) \right\},$$

and loss L is the mean squared error.

1. (1 pt) Calculate L.
   Ans:
2. (3 pt) Calculate partial derivatives of L with respect to the weights of the multilayer perceptron.
   Ans:
3. (1 pt) As in gradient descent, update the weights using step size equal to 1.
   Ans: