CSE7850/CX4803 - Spring 2024 - Homework 2

1 Machine Learning Basics [12 pts]

(1) [6pts] Consider the polynomial regression we discussed in our lecture, in which we have the independent variable (feature) x and the dependent variable (target) y and fit a polynomial of degree p, i.e., $y = \theta_p x^p + \theta_{p-1} x^{p-1} + \cdots + \theta_1 x + \theta_0$. How will the training, test, and validation errors change as p increases from 0?

(2) [6pts] Imagine you are a reviewer for a scientific journal that has received submissions comparing the performance of two machine learning algorithms, A and B, with respective hyperparameters α and β. Each submission makes a claim about the relative performance of these algorithms under certain evaluation settings. Your task is to assess these claims based on the provided evaluation settings and decide whether to recommend acceptance or rejection of the paper.

(a) *Claim*: Algorithm A outperforms Algorithm B.
Evaluation Setting: The optimal value of hyperparameter α for Algorithm A was selected by 10-fold cross-validation on the training set, while the default value was used for hyperparameter β of Algorithm B.

(b) *Claim*: Algorithm A outperforms Algorithm B.
Evaluation Setting: The performance comparison is based on test errors, with hyperparameters for both algorithms determined through 10-fold cross-validation on the training set.

(c) *Claim*: Algorithm A outperforms Algorithm B.
Evaluation Setting: The results reported correspond to the choices of hyperparameters α for Algorithm A and β for Algorithm B that each produces the best test error.

For each scenario, consider the fairness and rigor of the evaluation process described. For each claim and its associated setting listed below, provide a recommendation of "accept" or "reject" and a one-sentence justification for your decision.

2 Gradient Descent [13 pts]

Logistic regression is one of the widely-used model for classification. The logistic regression model $h_{\boldsymbol{\theta}}(x)$ has the form

$$\hat{y} = h_{\boldsymbol{\theta}}(x) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}, \tag{1}$$

where the vector $\boldsymbol{x}$ is the input, $\hat{y}$ is the output predicted by the model, and the vector $\boldsymbol{\theta}$ is the weights of the model. We simplify the setting such that there is only one training data point $(\boldsymbol{x}, y)$. Note that here $\boldsymbol{x} = [x_1, x_2, \ldots, x_m]^T$ is a m-dimensional features vector of the input and $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_m]^T$ is a vector of weights.

(1) [1pt] In class, we have seen the cross-entropy loss, a common loss function for classification problems. Write down this loss $L(y, \hat{y})$ in terms of y and $\hat{y}$.

(2) [4pts] The logistic regression is closely related to the sigmoid function, which is given by

$$g(z) = \frac{1}{1+e^{-z}}. \tag{2}$$

It is easy to see that the logistic model $h_{\boldsymbol{\theta}}(x)$ can be expressed in terms of the sigmoid function: $h_{\boldsymbol{\theta}}(x) = g(\boldsymbol{\theta}^T \boldsymbol{x})$. When running the backpropagation algorithm to learn the weights of the logistic model, we need to compute the derivative of the simoid function $g'(z)$. Show that

$$g'(z) = g(z)(1 - g(z)) \tag{3}$$

(3) [3pts] Show that $1 - g(z) = g(-z)$.

(4) [4pts] Derive $\nabla_{\theta_j} L$, i.e., the gradient of the loss with respect to θ_j. (Hint: you may what to apply the chain rule and reuse the fact in Eq. 3).

(5) [1pt] After backpropagation, we can use the computed gradients to update every parameter θ_j ($j = 1, \ldots, m$) using gradient descent. Write down the update step of gradient descent for θ_j.

3 Number of Optimal Alignments [15 pts]

From our lecture slides or HW1, you may have noticed that for a given input pair of sequences, the optimal alignment is not unique, i.e., there may exist more than one alignment that gives the optimal score. Please give an efficient algorithm that computes the total number of optimal alignments. (Note: you are expected to give an algorithm that is more efficient than the brute-force exhaustive search.) *Hint: use dynamic programming!*

4 Design a Neural Network by Hand [10 pts]

You have seen in our lectures that the non-linearity and multi-layer structure of a neural network increase the capacity to model complex data. For example, a neural network is able to classify points which cannot be perfectly classified by a *linear* classifier. In this problem, you will need to design a neural network that is more "powerful" than a linear classifier.

x_1	x_2	y
1	1	-1
1	0	+1
0	1	+1
0	0	-1

Table 1: Input data

Suppose you a given four points on a 2D plane, $\{x_1^{(i)}, x_2^{(i)}\}_{i=1}^4$. Each point is associated with a binary label $y^{(i)} \in \{-1, +1\}$, as shown in Table 1. First plot these points and convince yourself that there is no linear classifier (linear line) that can perfectly classify the four points. Now we want to show that a simple

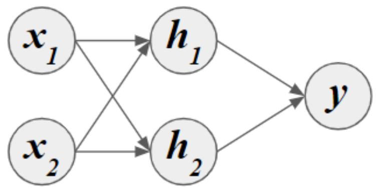

Figure 1: One-layer neural network

neural network, even with only one layer of hidden units, can correctly classify those points. Consider the neural network in Figure 1, which has 2 inputs (x_1 and x_2), 2 hidden nodes (h_1 and h_2), and one output. The network can be summarized by the following functions.

$$\begin{aligned} z_1 = h_1(x_1, x_2) &= \texttt{sgn}(W_{11}x_1 + W_{12}x_2 + b_1) \\ z_2 = h_2(x_1, x_2) &= \texttt{sgn}(W_{21}x_1 + W_{22}x_2 + b_2) \\ o &= \texttt{sgn}(W_{31}z_1 + W_{32}z_2 + b_3), \end{aligned}$$

where W_{k1}, W_{k2}, and b_k ($k = 1, 2, 3$) are the weights and biases of the neural network. Here, we use the sign function `sgn` as the activation function ($\texttt{sgn}(x)$ returns +1 if x is positive, and -1 otherwise). Please find a setting of all parameters (W_{k1}, W_{k2}, and b_k, $k = 1, 2, 3$) for this neural network that will perfect classify the four points.

Submission: Please use a text file "`weights.txt`" to save your answers of this question. There should be nine numbers in the file, separated by spaces (in one line). The order is $W_{11}, W_{12}, b_1, W_{21}, W_{22}, b_2, W_{31}, W_{32}, b_3$. Compress the file `weights.txt` together with other code files that need to be submitted into a zip and upload it to gradescope HW2-Code part.

5 Programming: Classification [25 pts]

For the programming problems, there is a Google Colab notebook link for each problem. You can either download it and run it locally or make a copy and run it on the Google Colab notebook.

Please see the statement of this question in the following Google Colab:
`https://colab.research.google.com/drive/15x8WdQ62len_au31zSsOFMoAd-93xeyZ?usp=sharing`

6 Programming: Neural Network [25 pts]

Please see the statement of this question in the following Google Colab:
`https://colab.research.google.com/drive/1A6oDBX3XXd_8jps5HXrtcheLq-lzPrt8?usp=sharing`

Submission instructions

For the first part, you will need to write the answers to Q1-3 in a PDF file. Please submit this PDF file to the "HW2-pdf" assignment in Gradescope.

For the second part, you will need to compress all of the answers or codes for the remaining questions into a single zip file. Please make sure to compress the files directly instead of compressing the root directly. Once you have created the zip file, please submit it to the "HW2-Code" assignment in Gradescope.

Please note that it is important to follow these instructions carefully to ensure that your submission is properly recorded and graded. If you have any questions or concerns about the submission process, please reach out to your TA or the course instructor. Have fun with this assignment!