CS 475 Machine Learning: Homework 3 Analytical (35 points)

Assigned: Friday, September 24, 2021
Due: Friday, October 8, 2021, 11:59 pm US/Eastern

Partner 1: Harrison Khoo (hkhoo2), Partner 2: Dimitri Lezcano (dlezcan1)

Instructions

We have provided this LaTeX document for turning in this homework. We give you one or more boxes to answer each question. The question to answer for each box will be noted in the title of the box. You can change the size of the box if you need more space.

Other than your name, do not type anything outside the boxes. Leave the rest of the document unchanged.

Do not add text outside of the answer boxes. You are allowed to make boxes larger if needed.

We strongly recommend you review your answers in the generated PDF to ensure they appear correct. We will grade what appears in the answer boxes in the submitted PDF, NOT the original latex file.

1 Neural Networks

1. Consider a neural network model with n layers, where the vertex for each internal layer is determined by a ReLU activation function applied to a weighted combination of vertices of the previous layer, while the output layer is the given by a weighted linear function of the outputs of the previous layer (without ReLU).

   (a) Show that if we replace all ReLU units by the identity function $f(x) = x$, the resulting neural network yields a linear regression model of the inputs.

   (b) Is there a unique loss minimizer setting of weights for this model? Explain.

(a) Suppose the activation function $f(x) = x$. Define the weights of each layer i as $\beta_i \in \mathbb{R}^{n_i \times n_{i+1}}$ where n_i is the input size of layer i, including bias term, (i.e. $n_1 = \dim(x_0) + 1$). Define $\bar{x} = (y^T, 1)^T$ and $\bar{x}_i = f(\beta_i \bar{x}_{i-1})$, using matrix multiplication of β_i and $\bar{x}_{i-1}$. Therefore, $\bar{x}_i = \beta_i \bar{x}_{i-1} \because f(x) = x$. Therefore, applying inductive reasoning, we have that the output, $\bar{x}_n$, is given by

$$\bar{x}_n = \left(\prod_{i=1}^{n} \beta_i \right) \bar{x}_0 = \beta \bar{x}_0$$

where $\beta = \prod_{i=1}^{n} \beta_i \in \mathbb{R}^{(\dim(x_0)+1) \times \dim x_n}$. Therefore, fitting outputs $y_i \in \mathbb{R}^{\dim x_n}$, we can formulate fitting the neural network to

$$\arg\min_{\beta} \sum_i ||y_i - \beta x_{0,1}||^2$$

which is of the form of a linear regression model.

(b) Using the formulation of (a), there is a unique loss minimizer setting using linear regression to determine β. However, in the general scenario where we use a non-linear activation (i.e. ReLU), there is not a unique minimizer loss setting for realizing the weights of the model. Given the non-linearities, there are increasing amounts of local minima in loss functions. Also, these weights will be optimized differently based on the different loss functions used (L1, L2, KL-Divergence, etc. will all vary differently). The parameters of these models are usually much larger than the dimension of the problem they are trying to fit and task-specific losses and intuition are needed to pick a good performing loss function, however, none of these settings will be unique.

2. Consider a neural network with a single intermediate layer and a single output, where the the activation function for every vertex in the intermediate layer is the sigmoid function $\sigma(v) = \frac{1}{1+\exp\{-v\}}$ of a weighted linear combination of inputs, while the output is a weighted linear combination of the all the intermediate vertices, and all weights are non-negative. Show that this model has a likelihood convex with respect to all weight parameters.

   Hints:

   - First show that the negative log likelihood for each intermediate layer vertex v (viewed as the outcome), with the weighted combination of inputs for that vertex (viewed as a single feature x) is a convex function. That is, show that:

     $$h(x) = -x \log \sigma(x) - (1-x)\log(1-\sigma(x))$$

     is convex in x. You can do this by showing that the second derivative of this function with respect to x is always non-negative.
   - Then, show that a composition of a linear function $g(w_1, \ldots, w_k)$ of several arguments and a convex function $h(x)$ that is convex in its single argument x is convex in each argument. That is, show that $h(g(w_1, \ldots, w_k))$ is convex in each $w_1, \ldots, w_k$.
   - Finally, show that a weighted combination of convex functions is also convex, if all weights are non-negative.

(a) First, note that $\frac{d\sigma(x)}{dx} = -\frac{\exp(-x)}{(1+\exp(-x))^2} = -\sigma(x)(1-\sigma(x))$. So we have that $\frac{d\log\sigma(x)}{dx} = \frac{1}{\sigma(x)}\frac{d\sigma(x)}{dx} = \sigma(x) - 1$. Furthermore, $\frac{d\log(1-\sigma(x))}{dx} = \frac{1}{1-\sigma(x)}\frac{d(-\sigma(x))}{dx} = \sigma(x)$. Thus,

$$\begin{aligned}\frac{d^2h}{dx^2} &= \frac{d}{dx}\left(-\log\sigma(x) + x(1-\sigma(x)) + \log(1-\sigma(x)) - (1-x)\sigma(x)\right)\\ &= (1-\sigma(x)) + 1 + \sigma(x) + \sigma(x)(1-\sigma(x)) > 0\ \forall x \in \mathbb{R}\\ &\implies h(x) \text{ is convex in } x\end{aligned}$$

(b) Second, if $h(x)$ is convex, then $\frac{d^2h}{dx^2} > 0\ \forall x$. Consider $g(w_1, \ldots, w_k)$ a linear function,

$$\frac{\partial^2 h(g(w_1,\ldots,w_k))}{\partial w_i^2} = \frac{\partial^2 h(g)}{\partial g^2}\left(\frac{\partial g}{\partial w_i}\right)^2 + \frac{\partial h}{\partial g}\frac{\partial^2 g}{\partial w_i^2} = \frac{\partial^2 h(g)}{\partial g^2}\left(\frac{\partial g}{\partial w_i}\right)^2 \because g \text{ is linear in } w_i$$

Therefore, $\because$ h is convex and $\frac{\partial g}{\partial w_i} \in \mathbb{R}$, $\frac{\partial^2 h(g)}{\partial w_i^2} > 0\ \forall w_i$. So $h(g)$ is convex in $w_1, \ldots, w_k$.

(c) Consider $h(x) = \sum_{i=1}^n c_i h_i(x)$ for $c_i > 0$ and h_i convex. Then $\frac{d^2h}{dx^2} = \sum_{i=1}^n c_i \frac{d^2h_i}{dx^2} > 0 \because c_i > 0$ & h_i is convex.

The negative log-likelihood of each of the vertices in the intermediate layer can be represented as $h_i(g_i(w_{i,1}, \ldots, w_{i,k}))$ where h_i is of the form in (a) and g_i is a linear function of the weights. By (b), $h_i(g_i(w_{i,1}, \ldots, w_{i,k}))$ is convex w.r.t $w_{i,j}\ \forall j = 1, \ldots, k$. Therefore, since the output is a sum of the intermediate vertices with non-negative weights, we have that the negative log-likelihood can be represented as a weighted sum of convex functions (w.r.t the parameters of the model) with non-negative weights. Therefore, by (c), the negative log-likelihood of the model is convex with respect to its parameters.

3. Assume there exist a setting of parameters for this model that minimizes the squared loss, such that at least two intermediate nodes have different weights feeding into their activation function. Show that there exists more than one setting of parameters that minimizes the squared loss. (This shows the parameters of this model are not identified).

Suppose that we have a set of weights where $\exists i, j$ such that $i \neq j$ and $w_i \neq w_j$ that minimizes squared loss. This means that we have

$$\frac{1}{2}\left(y - \sigma\left(\sum_{k=0}^{n} w_k x_k\right)\right)^2$$

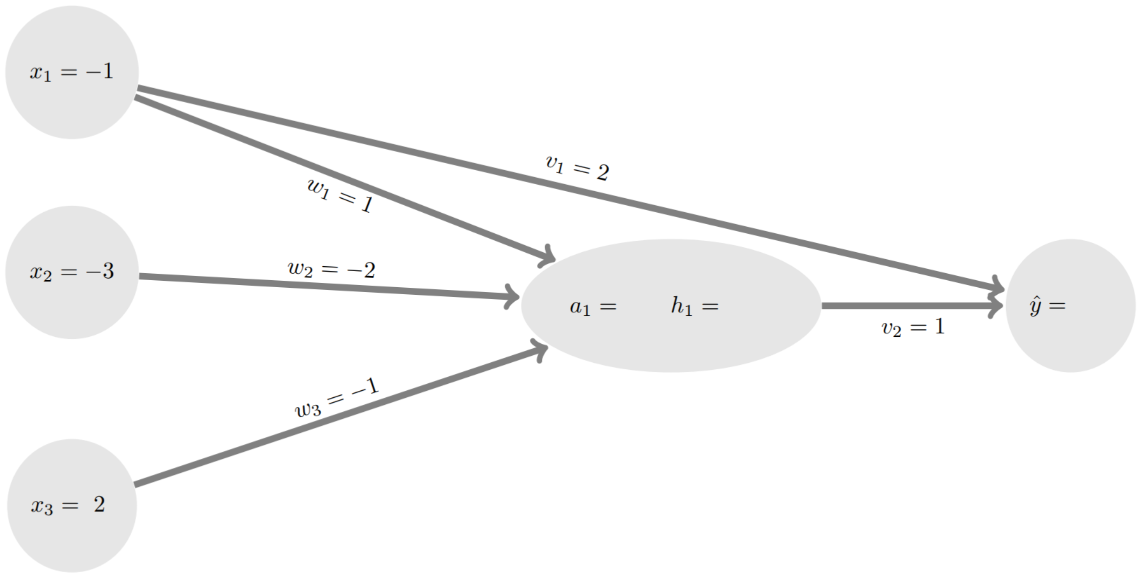
is minimized. Take $\delta \neq 0 \in \mathbb{R}$. Then let $w_i' = w_i + \delta x_j$ and $w_j' = w_j - \delta x_i$ and $w_k' = w_k \; \forall k \neq i, j$. Therefore,

$$\sum_{k=0}^{n} w_k' x_k = (w_i + \delta x_j)x_i + (w_j - \delta x_i)x_j + \sum_{k \neq i,j} w_k x_k = \sum_{k=0}^{n} w_k x_k$$

Thus we have another set of weights that produces $\sigma(\sum_k w_k' x_k) = \sigma(\sum_k w_k x_k)$ and therefore produces the same squared loss. Since the original parameters minimizes the squared loss, this new set of parameters also minimizes squared loss.

4. This question contains multiple parts. Please answer all of these parts in large the answer box following the question. (Feel free to increase the box for additional space.)

   Consider the 2-layer neural network shown below. There are three input features x_1, x_2, and x_3 which get fed into one activation (a_1) from which a hidden value is computed (h_1). The non-linearities connecting activations to hidden values are **rectified linear units** *ReLU*: $h_1 = ReLU(a_1)$, with $ReLU(x) = 0$ if $x < 0$, and $ReLU(x) = x$ otherwise. The output unit takes 2 inputs: the hidden value h_1 and x_1.



   (i) Execute forward propagation on this network, writing the appropriate values for a_1, a_2, h_1, h_2 and $\hat{y}$ in the figure above.

   (ii) Give the expression of $\hat{y}$ as a function of x_1, x_2, x_3, w_1, w_2, w_3, v_1, v_2 and the $ReLU(\cdot)$ function.

   (iii) The correct class for example $x = [x_1, x_2, x_3] = [-1, -3, -2]$ is $y = -1$. Please run the backpropagation algorithm to minimize the squared error loss $l = \frac{1}{2}(y - \hat{y})^2$ on this single example. Derive the mathematical expression of the gradients of the loss l with respect to weights w_1, w_2, and w_3, and calculate its numerical value.

   (iv) Indicate how the value of each parameter below changes after the update: does it increase, decrease, or stay the same?

   (v) Derive the update rule for parameters v_1 and v_2 when running the backpropagation algorithm on the same example x, with the squared loss l and a step size $\eta = \frac{1}{2}$. *Hint: you will need to (1) derive the appropriate gradient, (2) evaluate the gradient, (3) update your guess for the parameter β_{new} by subtracting the gradient times the step size from the old guess β_{old}.*

(i) $a_1 = -1 + 6 - 2 = 3$, $h_1 = 3$, $\hat{y} = h_1 + 2x_1 = 3 - 2 = 1$

(ii) $\hat{y} = ReLU(w_1x_1 + w_2x_2 + w_3x_3) + 2x_1$

(iii) The gradients of l w.r.t w_i is given by

$$\frac{\partial l}{\partial w_i} = \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial w_i} = -(y-\hat{y})\frac{\partial \hat{y}}{\partial w_i} = \begin{cases} -(y-\hat{y})x_i, & w_1x_1 + w_2x_2 + w_3x_3 > 0 \\ 0, & \text{otherwise} \end{cases}$$

We find that $\hat{y} = ReLU(-1 + 6 + 2) - 2 = 5$. So $(y - \hat{y}) = -1 - 5 = 6$. Therefore

$$\frac{\partial l}{\partial w_1} = -6(-1) = 6 \;\Bigg|\; \frac{\partial l}{\partial w_2} = -6(-3) = 18 \;\Bigg|\; \frac{\partial l}{\partial w_3} = -6(-2) = 12$$

(iv) In each case, we find that the update will follow the negative of the gradient direction and the update here $\Delta w_i \propto -\frac{\partial l}{\partial w_i} < 0 \;\forall i = 1, 2, 3$, so each parameter will decrease.

(v) $\frac{\partial l}{\partial v_1} = \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial v_1} = -(y-\hat{y})x_1 = -6(-1) = 6$ and $\frac{\partial l}{\partial v_2} = \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial v_2} = -(y-\hat{y})h_1 = -6(7) = -42$. $v_{i,new} = v_i - \eta\frac{\partial l}{\partial v_i}$, so we have $v_{1,new} = 2 - \frac{1}{2}(6) = -1$ and $v_{2,new} = 1 - \frac{1}{2}(-42) = 22$.

2 SVMs

1. Consider 2D classification problem with 3 classes with the following training sample:

   Class 1: (0,1), (0,2), (1,1)
   Class 2: (-2,0), (-1,0), (-1,-1)
   Class 3: (2,0), (3,0), (2,-1)

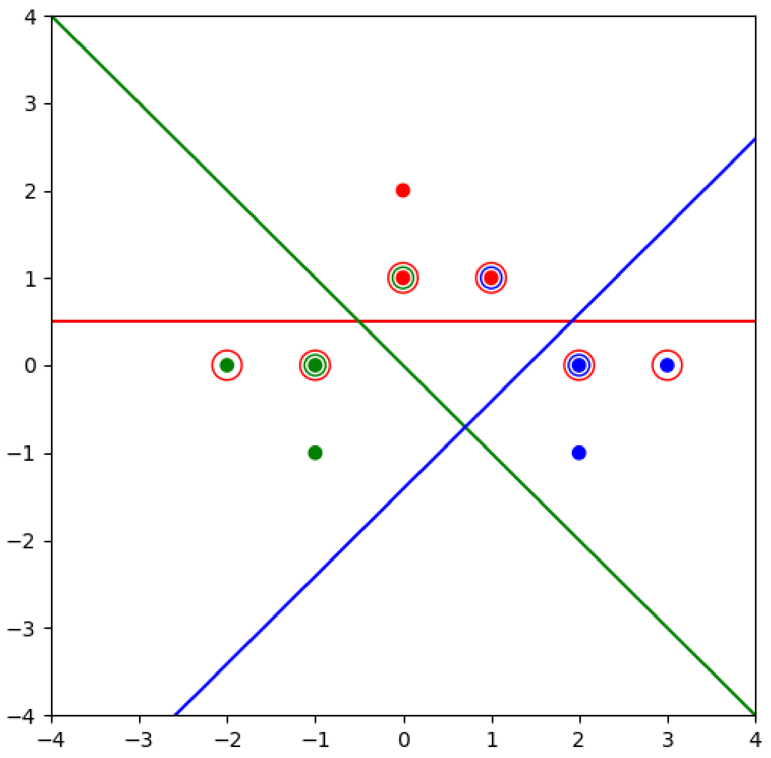
   (a) Explain how you would formulate an SVM model to solve this problem. (Hint: How many classifiers do you need?)

   We would need three classifiers to each of the different classes. We would have the following:

   $$\mathbb{I}_i(y)(w_i^T x + b_i) > 1, \; \forall i = 1, 2, 3$$

   Where $\mathbb{I}_i(y) = \begin{cases} +1, & y = i \\ -1, & y \neq i \end{cases}$. Therefore, we need to find 3 sets of (w_i, b_i) that classifies the distinction of Class i and not Class i for $i = 1, 2, 3$.

   (b) Find w, b, and corresponding support vectors for each classifier. Please draw the plot. You can take a screenshot or photo and use the `\includegraphics{}` directive.



(c) Given a test sample, explain how to make a decision.

Define each classifier for class i to be $f_i(x) = w_i^T x + b_i$, where the following are the parameters from the above SVM (support vectors circled with their corresponding color):

Class i	w_i	b_i
1	$(0, 1)$	$1/2$
2	$(-1/\sqrt{2}, -1/\sqrt{2})$	0
3	$(1/\sqrt{2}, -1\sqrt{2})$	1

Given a test sample, x_t, we can decide a class based on the highest class correspondence using

$$\hat{y}_t = \arg\max_i f_i(x_t) = \arg\max_i w_i^T x_t + b_i$$