STAT 542: Homework 4

Due: Mar. 24 midnight on Canvas

Please make sure that your solutions are readable and the file size is reasonable. Typing the answers is highly encouraged.

Problem 1.

[2pts] Derive the forward and backward propagation equations for the cross-entropy loss function.

Hint: The gradient update equations for the sum of squares loss can be found in p396 of Elements of Statistical Learning*, which is used for regression tasks. Here we consider the classification task and replace the sum of squares loss by the cross-entropy loss. You should highlight in your solution what parts have changed.*

Problem 2.

[2pts] In a K-class classification problem, suppose that for given neural network parameter (weights) θ and input x, the output of the network are functions $f_k(x, \theta)$, $k = 1, \dots,$ which are nonnegative numbers that sum to 1. Show that minimizing the cross-entropy loss is equivalent to maximum likelihood estimation, if $f_k(x, \theta)$ is interpreted as the likelihood $P(Y = k|x, \theta)$.

Problem 3.

Train a one hidden layer neural network with scalar input and output $(X_i, Y_i)_{i=1}^n$. Consider a toy problem setting where the activation function degenerates to a linear (identity) function[1], so that for any input x, the network output is calculated as

$$z_j := \alpha_j x; \tag{1}$$

$$f(x) := \frac{1}{\sqrt{p}} \sum_{j=1}^{p} \beta_j z_j \tag{2}$$

where z_j's are neurons in the hidden layer and $(\alpha_j)_{j=1}^p$ and $(\beta_j)_{j=1}^p$ are the weights. Define the loss function $E := \sum_{i=1}^n (Y_i - f(X_i))^2$. Suppose that the gradient descent step size is

[1]In a neural net we have $z_j := \alpha_j \sigma(x)$ with some nonlinear activation $\sigma()$. Here we consider the case of linear $\sigma()$ to allow explicit formula of weights, but conclusions of the exercise can be generalized to smooth $\sigma()$.

small, so that we focus on the continuous-time version of backpropagation:

$$\frac{d\alpha_j}{dt} = -\frac{\partial E}{\partial \alpha_j}; \tag{3}$$

$$\frac{d\beta_j}{dt} = -\frac{\partial E}{\partial \beta_j}. \tag{4}$$

1. [2pt] Express the right sides of (3) and (4) in terms of X_i, α_j, β_j, and the residue $R_i := Y_i - f(X_i)$ ($i = 1, \dots, n$).

2. [1pt] Show that when training completes (reaches a stationary point), the neural net function $\hat{f}(x) = \frac{1}{\sqrt{p}} \sum_{j=1} \hat{\alpha}_j \hat{\beta}_j x$ coincides with the least squares estimator. *Hint: set the right sides of* (3) *and* (4) *to zero.*

3. [1pt bonus] Show that for each j, $\alpha_j^2 - \beta_j^2$ remains a constant during training. *Hint: take the ratio of* (3) *to* (4) *and use part 1).*

4. [1pt bonus] Let $\bar{\alpha}_j$ and $\bar{\beta}_j$, $j = 1, \dots, p$ be weights at initialization. Show that $\hat{\alpha}_j + \hat{\beta}_j = c(\bar{\alpha}_j + \bar{\beta}_j)$ for some constant $c > 0$ independent of j. *Hint: take the sum of* (3) *and* (4) *and use part 1).*

5. [0.5pt bonus] Use previous parts to show $\hat{\alpha}_j - \hat{\beta}_j = \frac{1}{c}(\bar{\alpha}_j - \bar{\beta}_j)$.

6. [1pt bonus] Show that c must satisfy

$$\frac{1}{4\sqrt{p}} \sum_j \left[c^2(\bar{\alpha}_j + \bar{\beta}_j)^2 - \left(\frac{\bar{\alpha}_j - \bar{\beta}_j}{c} \right)^2 \right] = \frac{\sum_{i=1}^n Y_i X_i}{\sum_{i=1}^n X_i^2} \tag{5}$$

7. [1pt bonus] Suppose that the right side of (5) computed from the training data is 2, while the initialization $\bar{\alpha}_j, \bar{\beta}_j \sim \mathcal{N}(0,1)$ are iid. Show that for large p, $f(x)$ changes from approximately $\mathcal{N}(0, x^2)$ to $2x$ during training.

8. [1pt bonus] In the setting of Part 7), show that $c \to 1$ as $p \to \infty$. Note that this implies each α_j and β_j does not change much during training. Does it contradict the fact that $f(x)$ changes noticeably (from $\mathcal{N}(0, x^2)$ to $2x$)?

9. [1pt bonus] Suppose that $\bar{\alpha}_j, \bar{\beta}_j \sim \mathcal{N}(1,1)$ instead, and we replace (2) with a different normalization $f(x) := \frac{1}{p} \sum_{j=1}^p \beta_j z_j$. Show that c converges (as $p \to \infty$) to some constant not equal to 1.

Remark: In Part 8) we showed that each weight changes very little during training, implying that we can actually linearize the activation function so that the conclusion holds for general smooth activation (with a more involved analysis). This is the idea of neural tangent kernel `https://en.wikipedia.org/wiki/Neural_tangent_kernel`

Part 9) concerns the law of large numbers regime rather than CLT regime, and is often called the mean-field limit of the network.