

## Problem Set #3: Deep Learning & Unsupervised Learning

### Problem 1 A simple neural network

Let  $X = \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$  be dataset of  $m$  examples with 2 features. That is,  $x^{(i)} \in \mathbb{R}^2$ . Samples are classified into 2 categorie with labels  $y \in \{0, 1\}$ , as shown in Figure 1. Want to perform binary classification using a simple neural networks with the architecture shown in Figure 2.

Two features  $x_1$  and  $x_2$ , the three neurons in the hidden layer  $h_1, h_2, h_3$ , and the output neuron as  $o$ . Weight from  $x_i$  to  $h_j$  be  $w_{i,j}^{[1]}$  for  $i = 1, 2$  and  $j = 1, 2, 3$ , and weight from  $h_j$  to  $o$  be  $w_j^{[2]}$ . Finally, denote intercept weight for  $h_j$  as  $w_{0,j}^{[1]}$  and the intercept weight for  $o$  as  $w_0^{[2]}$ . Use average squared loss instead of the usual negative log-likelihood:

$$l = \frac{1}{m} \sum_{i=1}^m (o^{(i)} - y^{(i)})^2.$$

(a) Suppose we use sigmoid function as activation function for  $h_1, h_2, h_3$ , and  $o$ . We have

$$h_1 = g(w_1^{[1]}x), \quad h_2 = g(w_2^{[1]}x), \quad h_3 = g(w_3^{[1]}x), \quad o = g(w^{[2]}h).$$

Hence,

$$\frac{\partial l}{\partial w_{1,2}^{[1]}} = \frac{1}{m} \sum_{i=1}^m 2(o^{(i)} - y^{(i)})o^{(i)}(1 - o^{(i)})w_2^{[2]}h_2^{(i)}(1 - h_2^{(i)})x_1^{(i)},$$

where  $h_2^{(i)} = g(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1^{(i)} + w_{2,2}^{[1]}x_2^{(i)})$  and  $g$  is the sigmoid function. Therefore, the gradient descent update to  $w_{1,2}^{[1]}$ , assuming learning rate  $\alpha$  is

$$w_{1,2}^{[1]} := w_{1,2}^{[1]} - \frac{2\alpha}{m} \sum_{i=1}^m (o^{(i)} - y^{(i)})o^{(i)}(1 - o^{(i)})w_2^{[2]}h_2^{(i)}(1 - h_2^{(i)})x_1^{(i)}$$

where  $h_2^{(i)} = g(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1^{(i)} + w_{2,2}^{[1]}x_2^{(i)})$ .

(b) Now, suppose the activation function for  $h_1, h_2, h_3$ , and  $o$  is the step function  $f(x)$ , defined as

$$f(x) = \begin{cases} 1, & (x \geq 0), \\ 0, & (x < 0). \end{cases}$$

Is it possible to have a set of weights that allow the neural network to classify this dataset with 100% accuracy? If so, provide a set of weights by completing `optimal_step_weights` within `src/p01_nn.py` and explain your reasoning for those weights. If not, please explain the reasoning.

There is a set of weights that allow the neural network to classify this dataset with 100% accuracy. For the step function activation, we have

$$\begin{aligned} h_1 &= f(w_1^{[1]}x) = f(w_{0,1}^{[1]} + w_{1,1}^{[1]}x_1 + w_{2,1}^{[1]}x_2) \\ h_2 &= f(w_2^{[1]}x) = f(w_{0,2}^{[1]} + w_{1,2}^{[1]}x_1 + w_{2,2}^{[1]}x_2) \\ h_3 &= f(w_3^{[1]}x) = f(w_{0,3}^{[1]} + w_{1,3}^{[1]}x_1 + w_{2,3}^{[1]}x_2) \\ o &= f(w^{[2]}h) = f(w_0^{[2]} + w_1^{[2]}h_1 + w_2^{[2]}h_2 + w_3^{[2]}h_3). \end{aligned}$$

Notice from Figure 1 that the label  $y^{(i)} = 0$  if and only if  $x^{(i)}$  satisfies

$$\begin{cases} x_2^{(i)} > 0.5, \\ x_1^{(i)} > 0.5, \\ x_1^{(i)} + x_2^{(i)} < 4. \end{cases}$$

Now, let

$$w_1^{[1]} = \begin{bmatrix} 0.5 \\ 0 \\ -1 \end{bmatrix}, \quad w_2^{[1]} = \begin{bmatrix} 0.5 \\ -1 \\ 0 \end{bmatrix}, \quad w_3^{[1]} = \begin{bmatrix} -4 \\ 1 \\ 1 \end{bmatrix}, \quad w_1^{[2]} = \begin{bmatrix} -0.5 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

This set of weights will capture all the conditions and allow the neural network to classify this dataset with 100% accuracy.

- (c) Let the activation function for  $h_1$ ,  $h_2$ ,  $h_3$ , and  $o$  is the linear function  $f(x) = x$ , and the activation function for  $o$  be the same step function as before. Is it possible to have a set of weights that allow the neural network to classify this dataset with 100% accuracy? If so, provide a set of weights by completing `optimal_linear_weights` within `src/p01_nn.py` and explain your reasoning for those weights. If not, please explain the reasoning. ■

**Problem 2 KL divergence and maximum likelihood**

Kullback-Leibler (KL) divergence is a measure of how much one probability distribution is different from a second one. The *KL divergence* between two discrete-valued distribution  $P(X)$ ,  $Q(X)$  over the outcome space  $\mathcal{X}$  is defined as follows:

$$D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}.$$

Assume  $P(x) > 0$  for all  $x$ . (One other standard thing to do is adopt the convention that  $0 \log 0 = 0$ .) Sometimes, we also write the KL divergence more explicitly as  $D_{\text{KL}}(P \parallel Q) = D_{\text{KL}}(P(X) \parallel Q(X))$ .

*Background on Information Theory*

The *entropy* of a probability distribution  $P(X)$ , defined as

$$H(P) = - \sum_{x \in \mathcal{X}} P(x) \log P(x).$$

measures how dispersed a probability distribution is. Notably,  $\mathcal{N}(\mu, \sigma^2)$  has the highest entropy among all possible continuous distribution that has mean  $\mu$  and variance  $\sigma^2$ . The entropy  $H(P)$  is the best possible long term average bits per message (optimal) that can be achieved under probability distribution  $P(X)$ .

The *cross entropy* is defined as

$$H(P, Q) = - \sum_{x \in \mathcal{X}} P(x) \log Q(x).$$

The cross entropy  $H(P, Q)$  is the long term average bits per message (suboptimal) that results under a distribution  $P(X)$ , by reusing an encoding scheme designed to be optimal for a scenario with probability distribution  $Q(X)$ .

Notice that

$$D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log P(x) - \sum_{x \in \mathcal{X}} P(x) \log Q(x) = H(P, Q) - H(P).$$

If  $H(P, Q) = 0$ , then it necessarily means  $P = Q$ . In ML, it is common task to find distribution  $Q$  that is close to another distribution  $P$ . To achieve this, we optimize  $D_{\text{KL}}(P \parallel Q)$ . Later we will see that Maximum Likelihood Estimation turns out to be equivalent minimizing KL divergence between the training data and the model.

(a) **Nonnegativity.** Prove that

$$D_{\text{KL}}(P \parallel Q) \geq 0$$

and  $D_{\text{KL}}(P \parallel Q) = 0$  if and only if  $P = Q$ .

**Hint:** Use Jensen's inequality.

*Proof.* By definition,

$$D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} = - \sum_{x \in \mathcal{X}} P(x) \log \frac{Q(x)}{P(x)}.$$

Since  $-\log x$  is strictly convex, by Jensen's inequality, we have

$$D_{\text{KL}}(P \parallel Q) = - \sum_{x \in \mathcal{X}} P(x) \log \frac{Q(x)}{P(x)} \geq - \log \sum_{x \in \mathcal{X}} P(x) \frac{Q(x)}{P(x)} = 0.$$

When the equality holds,

$$\log \frac{Q(x)}{P(x)} = 0$$

with probability 1. That is,  $Q = P$  with probability 1. This completes the proof.  $\square$

- (b) **Chain rule for KL divergence.** The KL divergence between 2 conditional distributions  $P(X \mid Y)$ ,  $Q(X \mid Y)$  is defined as follows:

$$D_{\text{KL}}(P(X \mid Y) \parallel Q(X \mid Y)) = \sum_y P(y) \left( \sum_x P(x \mid y) \log \frac{P(x \mid y)}{Q(x \mid y)} \right).$$

This can be thought of as the expected KL divergence between the corresponding conditional distributions on  $x$ . That is, between  $P(X \mid Y = y)$  and  $Q(X \mid Y = y)$ , where the expectation is taken over the random  $y$ .

Prove the following chain rule for KL divergence:

$$D_{\text{KL}}(P(X, Y) \parallel Q(X, Y)) = D_{\text{KL}}(P(X) \parallel Q(X)) + D_{\text{KL}}(P(Y \mid X) \parallel Q(Y \mid X)).$$

*Proof.*

$$\begin{aligned} \text{LHS} &= \sum_x \sum_y P(x, y) \log \frac{P(x, y)}{Q(x, y)} \\ &= \sum_x \sum_y P(y \mid x) P(x) \left[ \log \frac{P(y \mid x)}{Q(y \mid x)} + \log \frac{P(x)}{Q(x)} \right] \\ &= \sum_x \sum_y P(y \mid x) P(x) \log \frac{P(y \mid x)}{Q(y \mid x)} + \sum_x P(x) \log \frac{P(x)}{Q(x)} \sum_y P(y \mid x) \\ &= \sum_x \sum_y P(y \mid x) P(x) \log \frac{P(y \mid x)}{Q(y \mid x)} + \sum_x P(x) \log \frac{P(x)}{Q(x)} \\ &= D_{\text{KL}}(P(X) \parallel Q(X)) + D_{\text{KL}}(P(Y \mid X) \parallel Q(Y \mid X)) \\ &= \text{RHS}. \end{aligned}$$

$\square$

- (c) **KL and maximum likelihood.** Consider density estimation problem and suppose we are given training set  $\{x^{(i)}\}_{i=1}^m$ . Let the empirical distribution be  $\hat{P}(x) = \frac{1}{m} \sum_{i=1}^m \mathbb{I}\{x^{(i)} = x\}$ . ( $\hat{P}$  is just the uniform distribution over the training set; i.e., sampling from the empirical distribution is the same as picking a random example from the training set.)

Suppose we have a family of distributions  $P_\theta$  parametrized by  $\theta$ . Prove that finding the maximum likelihood estimates for the parameter  $\theta$  is equivalent to finding  $P_\theta$  with minimal KL divergence from  $\hat{P}$ . That is, prove that

$$\operatorname{argmin}_{\theta} D_{\text{KL}}(\hat{P} \parallel P_{\theta}) = \operatorname{argmax}_{\theta} \sum_{i=1}^m \log P_{\theta}(x^{(i)}).$$

*Proof.* Notice that  $\hat{P}$  is the uniform distribution over the training set, thus  $\hat{P}(x^{(i)}) = \frac{1}{m}$  for  $i = 1, \dots, m$ . It follows that

$$D_{\text{KL}}(\hat{P} \parallel P_{\theta}) = \sum_x \hat{P}(x) \log \frac{\hat{P}(x)}{P_{\theta}(x)} = -\log m - \frac{1}{m} \sum_{i=1}^m \log P_{\theta}(x^{(i)}).$$

Hence,

$$\operatorname{argmin}_{\theta} D_{\text{KL}}(\hat{P} \parallel P_{\theta}) = \operatorname{argmax}_{\theta} \sum_{i=1}^m \log P_{\theta}(x^{(i)}),$$

as desired. □

**Remark:** Consider the relationship between parts (b-c) and multi-variate Bernoulli Naive Bayes parameter estimation. In Naive Bayes model we assumed  $P_{\theta}$  is the following form:  $P_{\theta}(x, y) = p(y) \prod_{i=1}^n p(x_i | y)$ . By the chain rule for KL divergence, we therefore have

$$D_{\text{KL}}(\hat{P} \parallel P_{\theta}) = D_{\text{KL}}(\hat{P}(y) \parallel p(y)) + \sum_{i=1}^n D_{\text{KL}}(\hat{P}(x_i | y) \parallel p(x_i | y)).$$

This shows that finding the maximum likelihood/minimum KL divergence estimates of the parameters decomposes into  $2n + 1$  independent optimization problems: One for the class priors  $p(y)$ , and one for each conditional distributions  $p(x_i | y)$  for each feature  $x_i$  given each of the two possible labels for  $y$ . Specifically, finding the maximum likelihood estimates for each of these problems individually results in also maximizing the likelihood of the joint distribution. This similarly applies to Bayesian networks. △

■