Homework 1
600.482/682 Deep Learning
Spring 2020

February 24, 2020

Due Sunday 2/23 11:59pm.
Please type your answers inline of the LaTeX file
Submit PDF to Gradescope with entry code 9G83Y7

1. In this exercise you are going to derive the well-known sigmoid expression for a Bernoulli distributed (binary) problem. The probability of the "positive" event occurring is p. The probability of the "negative" event occurring is $q = 1 - p$.

   (a) What are the odds o of the "positive" event occurring? Please express the result using p only.
   In statistics, the logit of the probability is the logarithm of the corresponding odds, i.e. $\text{logit}(p) = \log(o)$.

   (b) Given $\text{logit}(p) = x$, please derive the inverse function $\text{logit}^{-1}(x)$. Please express the result using x only.
   The inverse function of the logit in (b) is actually the sigmoid function $S(x)$. You may already have noticed that the probability $p = \text{logit}^{-1}(x) = S(x)$. This means that the range of the sigmoid function is the same as the range of a probability, i.e. $(0, 1)$. The domain of the sigmoid function is $(-\infty, \infty)$. Therefore, the sigmoid function maps all real numbers to the interval $(0, 1)$.

   (c) Now we look into the saturation of the sigmoid function. Calculate the value of the sigmoid function $S(x)$ for $x = \pm 100, \pm 10$, and 0. Round the results to two decimal places.

   (d) Calculate the derivatives of the sigmoid function $S'(x)$ and the value of $S'(x)$ for $x = \pm 100, \pm 10$, and 0. Round the results to two decimal places.
   You may have noticed that $S(\pm 100)$ is very close to $S(\pm 10)$; the derivatives at $x = \pm 100$ and $x = \pm 10$ are very close to zero. This is the saturation of the sigmoid function when $|x|$ is large. The saturation brings great difficulty in training deep neural networks. This will reappear in later lectures.

   Answer

   (a)

   $$o(p) = \frac{p}{1-p}$$

   (b)

   $$logit(x) = log(\frac{x}{1-x})$$
   $$y = log(\frac{x}{1-x})$$
   $$e^x = \frac{y}{1-y}$$
   $$f^{-1}(x) = \frac{e^x}{1+e^x}$$

(c)

$$\begin{aligned} S(100) &\approx 1.00 \\ S(-100) &\approx 0.00 \\ S(10) &\approx 1.00 \\ S(-10) &\approx 0.00 \\ S(0) &\approx 0.50 \end{aligned}$$

(d)

$$\frac{d}{dx}\left(\frac{1}{1+\exp(-x)}\right) = \frac{e^{-x}}{(e^{-x}+1)^2}$$

$$\begin{aligned} S'(100) &\approx 0.00 \\ S'(-100) &\approx 0.00 \\ S'(10) &\approx 0.00 \\ S'(-10) &\approx 0.00 \\ S'(0) &\approx 0.25 \end{aligned}$$

2. Recall in class, we learned the form of a linear classifier as $f(\boldsymbol{x}; \boldsymbol{W}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$. We will soon learn, that iteratively updating the weights in negative gradient direction will allow us to slowly move towards an optimal solution. We will call this technique backpropagation. Obviously, computing gradients is an important component of this technique. We will investigate the first derivative of a commonly used loss function: the softmax loss. Here, we consider a multinomial (multiple classes) problem.
Let's first define the notations:

$$\begin{aligned} \text{input features}: &\quad \boldsymbol{x} \in \mathbb{R}^D. \\ \text{target labels (one-hot encoded)}: &\quad \boldsymbol{y} \in \{0,1\}^K. \\ \text{multinomial linear classifier}: &\quad \boldsymbol{f} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \quad \boldsymbol{W} \in \mathbb{R}^{K \times D} \text{ and } \boldsymbol{f}, \boldsymbol{b} \in \mathbb{R}^K \\ \text{e.g., for the k-th classification}: &\quad f_k = \boldsymbol{w}_k^T \boldsymbol{x} + b_k, \text{ corresponding to } y_k, \\ &\quad \text{where } \boldsymbol{w}_k^T \text{ is the k-th row of } \boldsymbol{W}, k \in \{1...K\} \end{aligned}$$

   (a) Please express the softmax loss of logistic regression, $L(\boldsymbol{x}, \boldsymbol{W}, \boldsymbol{b}, \boldsymbol{y})$ using the above notations.

   (b) Please calculate its gradient derivative $\frac{\partial L}{\partial \boldsymbol{w}_k}$.

   Answer

   (a)

$$L(x, W, b, y) = -\log\left(\frac{e^{\boldsymbol{w}_i^T \boldsymbol{w} + b_i}}{\sum_{j=1}^K e^{\boldsymbol{w}_j^T \boldsymbol{x} + b_j}}\right)$$

   (b)

$$\frac{\partial L}{\partial w_k} == -\frac{\partial}{\partial \boldsymbol{w}_k}\left(\boldsymbol{w}_i^T \boldsymbol{x} + b_i\right) + \frac{\partial}{\partial \boldsymbol{w}_k} \log\left(\sum_{j=1}^k e^{\boldsymbol{w}_j^T \boldsymbol{x} + b_j}\right)$$

   for $y_k = 1$:
   $\frac{\partial L}{\partial \boldsymbol{w}_k} = \boldsymbol{x}^T \left(\frac{e^{\boldsymbol{w}_k^T \boldsymbol{x} + b_k}}{\sum_{j=1}^K e^{\boldsymbol{w}_j^T \boldsymbol{x} + b_j}} - 1\right)$ else:
   $\frac{\partial L}{\partial \boldsymbol{w}_k} = \boldsymbol{x}^T \frac{e^{\boldsymbol{w}_k^T \boldsymbol{x} + b_k}}{\sum_{j=1}^K e^{\boldsymbol{w}_j^T \boldsymbol{x} + b_j}}$

3. In class, we briefly touch upon the Kullback-Leibler (KL) divergence as another loss function to quantify agreement between two distributions p and q. In machine learning scenarios, one of these two distributions will be determine by our training data, while the other is being generated as output of our model. The goal of training our model is to match these two distributions as well as possible. KL divergence is asymmetric, so that assigning these distributions to p and q will matter. Here, you will investigate this difference by calculating the gradient. The KL divergence is defined as

$$\mathrm{KL}(p||q) = \sum_d p(d) \log\left(\frac{p(d)}{q(d)}\right)$$

   (a) Show that KL divergence is asymmetric using the following example. We define a discrete random variable X. Now consider the case that we have two sampling distributions $P(x)$ and $Q(x)$, which we present as two vectors that express the frequency of event x:

$$P(x) = [1,\ 6,\ 12,\ 5,\ 2,\ 8,\ 12,\ 4]$$
$$Q(x) = [1,\ 3,\ 6,\ 8,\ 15,\ 10,\ 5,\ 2]$$

   Please compute 1) the probability distribution, $p(x)$ and $q(x)$ (hint: calculate the normalization); and 2) both directions of KL divergence, $\mathbf{KL}(p||q)$ and $\mathbf{KL}(q||p)$.

   (b) Next, we try to optimize the weights $\boldsymbol{W}$ of a model in an attempt to minimize KL divergence. As a consequence, $q = q_{\boldsymbol{W}}$ now depends on the weights. Please express $\mathbf{KL}(q_{\boldsymbol{W}}||p)$ and $\mathbf{KL}(p||q_{\boldsymbol{W}})$ as optimization objective functions. Can you tell which direction is easier for computation? To find out, please look back at the original expression of $\mathbf{KL}(q_{\boldsymbol{W}}||p)$ and $\mathbf{KL}(p||q_{\boldsymbol{W}})$ and see which terms can be grouped to be a constant. This constant can be thus cancelled out when calculating the gradient. Then, please also calculate the gradient of $\mathbf{KL}(q_{\boldsymbol{W}}||p)$ and $\mathbf{KL}(p||q_{\boldsymbol{W}})$ w.r.t. $q_{\boldsymbol{W}}(d)$, the d-th element of $q_{\boldsymbol{W}}$.

   Answer

   (a) 1)
   p(x) = [0.02, 0.12, 0.24, 0.1, 0.04, 0.16, 0.24, 0.08]
   q(x) = [0.02, 0.06, 0.12, 0.16, 0.3, 0.2, 0.1, 0.04]
   2)

$$KL\left(p\|q\right) = 0.35$$

$$KL\left(q\|p\right) = 0.48$$

   (b) $\mathbf{KL}(p||q_{\boldsymbol{W}})$ should be easier for computation.

$$KL(qw||p) = \sum_d qw(d) * \log\frac{qw(d)}{p(d)}$$
$$KL(p||qw) = \sum_d p(d) * \log\frac{p(d)}{qw(d)}$$
$$= \sum_d p(d) * (\log p(d) - \log qw(d))$$

$$\frac{\partial KL(qw||p)}{\partial qw(d)} = \log qw(d) + 1 - \log p(d)$$
$$\frac{\partial KL(p||qw)}{\partial qw(d)} = \frac{p(d)}{qw(d)}$$

4. In this problem, you are provided an opportunity to perform hands-on calculation of the SVM loss and softmax loss we learned in class.

We define a linear classifier:

$$f(\boldsymbol{x}, \boldsymbol{W}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$$

and are given a data sample:

$$\boldsymbol{x}_i = \begin{bmatrix} -15 \\ 22 \\ -44 \\ 56 \end{bmatrix}, \; y_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Assume that the weights of our model are given by

$$\boldsymbol{W} = \begin{bmatrix} 0.01, & -0.05, & 0.1, & 0.05 \\ 0.7, & 0.2, & 0.05, & 0.16 \\ 0.0, & -0.45, & -0.2, & 0.03 \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} 0.0 \\ 0.2 \\ -0.3 \end{bmatrix}.$$

Please calculate 1) SVM loss (hinge loss) and 2) softmax loss (cross-entropy loss) of this sample. Use the natural log.