Machine Learning I: Foundations
Exercise Sheet 4

Prof. Marius Kloft TA:Billy Joe Franks

25.05.2020
Deadline: 26.05.2020

1) (MANDATORY) 10 Points

In this exercise, we will be deriving everything necessary for the gradient of a neural network. Later in the lecture you will see these terms in action. Calculate the following derivatives for $a, y \in \mathbb{R}, \mathbf{x}, \mathbf{b}, \mathbf{w} \in \mathbb{R}^d, W \in \mathbb{R}^{d \times d}$:

a) $\frac{\partial}{\partial W_{i,j}}(W\mathbf{x} + \mathbf{b})$.

$$\frac{\partial}{\partial W_{i,j}}(W\mathbf{x} + \mathbf{b}) = \frac{\partial}{\partial W_{i,j}} W\mathbf{x} = \frac{\partial}{\partial W_{i,j}} \begin{bmatrix} \sum_k W_{1,k} x_k \\ \vdots \\ \sum_k W_{i,k} x_k \\ \vdots \\ \sum_k W_{d,k} x_k \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ x_j \\ \vdots \\ 0 \end{bmatrix}$$

b) $\frac{\partial}{\partial b_i}(W\mathbf{x} + \mathbf{b})$.

$$\frac{\partial}{\partial b_i}(W\mathbf{x} + \mathbf{b}) = \frac{\partial}{\partial b_i}\mathbf{b} = \frac{\partial}{\partial b_i} \begin{bmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_d \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

c) $\frac{\partial}{\partial x_i}(W\mathbf{x} + \mathbf{b})$.

$$\frac{\partial}{\partial x_i}(W\mathbf{x} + \mathbf{b}) = \frac{\partial}{\partial x_i} W\mathbf{x} = \frac{\partial}{\partial x_i} \begin{bmatrix} \sum_k W_{1,k} x_k \\ \vdots \\ \sum_k W_{d,k} x_k \end{bmatrix} = \begin{bmatrix} W_{1,i} \\ \vdots \\ W_{d,i} \end{bmatrix}$$

d) $\frac{\partial}{\partial a}\sigma(a) = \frac{\partial}{\partial a}\left(1 + e^{-a}\right)^{-1}$.

$$\frac{\partial}{\partial a}\left(1+e^{-a}\right)^{-1} = \left(1+e^{-a}\right)^{-2} e^{-a}$$

In addition we note

$$\frac{e^{-a}}{(1+e^{-a})^2} = \sigma(a)\frac{1+e^{-a}-1}{1+e^{-a}} = \sigma(a)\left(1-\frac{1}{1+e^{-a}}\right) = \sigma(a)\left(1-\sigma(a)\right)$$

e) $\frac{\partial}{\partial a}\log\left(1+e^{-ya}\right)$.

$$\frac{\partial}{\partial a}\log\left(1+e^{-ya}\right) = \frac{-ye^{-ya}}{1+e^{-ya}} = -ye^{-ya}\sigma(ya)$$

f) Use the above to calculate $\frac{\partial}{\partial W_{i,j}}\log\left(1+\exp(-y\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))\right)$. $\hat{\sigma}$ is the elementwise application of σ, i.e.

$$\hat{\sigma}\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}\right) = \begin{bmatrix}\sigma(x_1)\\\sigma(x_2)\end{bmatrix}$$

In this exercise we will use the chain rule to simplify the derivation of derivatives.

$$\begin{aligned}
&\frac{\partial}{\partial W_{i,j}}\log\left(1+\exp(-y\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))\right)\\
=&\frac{\partial\log\left(1+\exp(-y\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))\right)}{\partial\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))}\frac{\partial\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))}{\partial\hat{\sigma}(W\mathbf{x}+\mathbf{b}))}\frac{\partial\hat{\sigma}(W\mathbf{x}+\mathbf{b}))}{\partial W\mathbf{x}+\mathbf{b}}\frac{\partial W\mathbf{x}+\mathbf{b}}{\partial W_{i,j}}\\
=&f_e(\mathbf{w}^T\hat{\sigma}(W\mathbf{x}+\mathbf{b}))\mathbf{w}^T\mathrm{diag}(\hat{f}_d(W\mathbf{x}+\mathbf{b}))\begin{bmatrix}0\\\vdots\\x_j\\\vdots\\0\end{bmatrix}
\end{aligned}$$

with

$$f_e(a) := -ye^{-ya}\sigma(ya)$$

$$f_d(a) := \sigma(a)\left(1-\sigma(a)\right)$$

And as above $\hat{f}_d(\mathbf{x})$ is the elementwise application of f_d to $\mathbf{x}$. We note that for optimization we would also have to calculate the derivative according to $\mathbf{b}$, however, this is very simple to adjust now. We just have to replace the last derivative with the one found in b).

2) We will explore activation functions for neural networks. For each function do the following, plot it, state the range of its output (i.e., $f(\mathbb{R})$), derive its gradient, and state an advantage and disadvantage of using it.

a) **TanH**: $f(x) = \tanh(x)$

b) **Error function**: $f(x) = \text{erf}(x)$

c) **ReLU**: $f(x) = \max(0, x)$

d) **Leaky-ReLU**: $f(x) = \max(0.01x, x)$

e) **PLU**: $f(x) = \max(\alpha(x + c) - c, \min[\alpha(x - c) + c, x])$

f) **Sinusoid**: $f(x) = \sin(x)$

g) **Gaussian**: $f(x) = e^{-x^2}$

3) The aim of this question is to show the similarities between the hinge loss and logistic loss in optimization. For the SVM problem we considered using gradient descent in slide 10 (Lecture 3.3). We proposed two such methods one using the hinge loss $h(\mathbf{x}_i) = \max(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i))$ and the other using the logistic loss function:

$$l(x_i) = \ln(1 + e^{-\mathbf{w}^\top \Phi(\mathbf{x}_i)}),$$

where $\Phi(\mathbf{x}_i)$ is the output of the output layer.

a) Let $a_1, a_2, \cdots, a_n \in \mathbb{R}^+$, where $a_i \neq a_j$, $a^* = \max(\{a_1, a_2, \cdots, a_n\})$ and $a^{**} = \max(\{a_1, a_2, \cdots, a_n\} \backslash \{a^*\})$. Show that:

$$\lim_{(a^{**}-a^*)\to-\infty} \ln(e^{a_1} + e^{a_2} + \cdots + e^{a_n}) = \max(\{a_1, a_2, \cdots, a_n\}) = a^*$$

$$\begin{aligned}
& \lim_{(a^{**}-a^*)\to-\infty} \ln(e^{a_1} + e^{a_2} + \cdots e^{a_n}) \\
= & \lim_{(a^{**}-a^*)\to-\infty} \ln(\frac{e^{a^*}}{e^{a^*}}(e^{a_1} + e^{a_2} + \cdots + e^{a^*} + \cdots + e^{a_n})) \\
= & \lim_{(a^{**}-a^*)\to-\infty} \ln(e^{a^*}(\frac{e^{a_1}}{e^{a^*}} + \frac{e^{a_2}}{e^{a^*}} + \cdots + \frac{e^{a^*}}{e^{a^*}} + \cdots + \frac{e^{a_n}}{e^{a^*}})) \\
= & \lim_{(a^{**}-a^*)\to-\infty} \ln(e^{a^*}(e^{a_1-a^*} + e^{a_2-a^*} + \cdots + e^{a^*-a^*} + \cdots + e^{a_n-a^*})) \\
=^1 & \ln(e^{a^*}(0 + 0 + \cdots + 1 + \cdots + 0)) \\
= & \ln e^{a^*} \\
= & a^* = \max\{a_1, a_2, \cdots, a_n\}
\end{aligned}$$

The equals sign with [1] is not reasonable as written. In fact it is multiple steps in one. Using the limit of compositions the limit is drawn past ln and then it is pulled into each summand, where the limit is evaluated. The reason for this simplification, is simply space of this sheet.

b) Using item (a), show that the hinge loss can be approximated asymptotically by the calculated limit.

From item (a) we have:

$$\max(a_1, a_2, \cdots, a_n) \quad = \quad \lim_{(a^{**}-a^{*})\to-\infty} \ln\left(e^{a_1} + e^{a^2} + \cdots + e^{a_n}\right)$$

.

We can write the hinge loss as:

$$h(t) = \max(0, t).$$

If $t \to \infty$ then $(0 - t) \to -\infty$.Therefore, asymptotcally

$$h(t) = \ln(e^0 + e^t)$$

In the SVM case:

$$\begin{aligned} h^*(t) &= \ln(e^0 + e^{1-y_i(\mathbf{w}^T x_i)}) \\ &= \ln(1 + e^{1-y_i(\mathbf{w}^T x_i)}) \end{aligned}$$

.

c) You probably noticed that the hinge loss can be approximated by the function $h^*(t) = \ln(1 + e^{1-t})$, where $t = y_i(\mathbf{w}^\top \mathbf{x}_i)$. Compare $h^*(t)$ to $l(t) = \ln(1 + e^{-t})$ (logistic loss).

Essentially $\ln(1+e^{1-t})$ and the hinge-loss fit each other perfectly. The logistic loss on the other hand is shifted to the right in relation to the hinge-loss. This exercise was supposed to show you that indeed the logistic loss is an approximation of the hinge-loss.

4) Solve programming task 4.