problems2

November 20, 2018

1 Problem sheet 2

1.1 1

Implement gradient descent for the linear model in `keras`. As in the lecture, use the functions `k_update_sub` and `k_gradients`. Gradually increase the difficulty: 1. Add biases. 2. Add a second layer. 3. Add ReLU activation function.

1.2 2

Refine your gradient descent algorithm from problem 1, so as to support *stochastic gradient descent*.

1.3 3

In this problem, we understand why weights need to be initialized at random when using gradient descent. 1. Show that if all entries of $W \in \mathbb{R}^{n\times p}$ are equal to a common value, then the function $x \mapsto Wx$ is invariant under permutation. That is, if $x = (x_1, \dots, x_p)^\top \in \mathbb{R}^p$ and π is a permutation of $\{1, \dots, p\}$, then $Wx = W(x_{\pi(1)}, \dots, x_{\pi(p)})^\top$. 2. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function that is invariant under permutation. Defining $g(W) = f(Wx)$, show that if $W_0 \in \mathbb{R}^{n\times p}$ is such that all its entries are equal to a common value, then

$$\frac{\partial}{\partial w_{ij}} g(W)\Big|_{W=W_0} = \frac{\partial}{\partial w_{i'j}} g(W)\Big|_{W=W_0}.$$

In particular, gradient descent would update $w_{i'j}$ and w_{ij} always by the same amount.

1.4 4

Explain why backpropagation computes gradients in top-to-bottom order. Why would it be unwise to proceed in a bottom-up order as in the forward pass?

1.5 5

Finish the computation of $\nabla_{W_1} f(x)$.

1.6 6

Would it not be better to choose a linear activation function $a(x) = c \cdot x$ instead of ReLU? Here, there is never a problem of a vanishing gradient?

1.7 7

Why don't we use the Newton-Raphson or other second-order algorithms instead of plain gradient descent?