Lab 3 - Artificial Neural Network

{bengio,mkeller}@idiap.ch
http://www.idiap.ch/~{bengio,mkeller}

December 7, 2005

1. **Download `data.py` and `mlp.py`. Choose a UCI database (*eg* `pi-diabetes`), split it in train, validation and test sets and train a Multi-Layers Perceptron, with and without normalizing the data. Try also different cost functions.**

2. **Show that to maximize the likelihood under the hypothesis that the observations y_l ($l \in \{1, \ldots, L\}$) are generated from a smooth function with added noise ξ following a Gaussian distribution $\mathcal{N}(0,1)$, $y_l = f_\theta(x_l) + \xi$, is equivalent to minimize the empirical risk with Mean Square Error function. (Hint: Consider $P_\theta(y_l|x_l)$).**

   ....

   The log-likelihood over the training set:

   $$\log \mathcal{L}(\theta) = \log(\prod_{l=1}^{L} P_\theta(y_l|x_l)) = \sum_{l=1}^{L} \log P_\theta(y_l|x_l).$$

   Given the hypothesis on the generation of the observation y_l, we have:

   $$P_\theta(y_l|x_l) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}\|y_l - f_\theta(x_l)\|^2),$$

   and thus:

   $$\log \mathcal{L}(\theta) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\sum_{l=1}^{L} \|y_l - f_\theta(x_l)\|^2$$

   ....

3. **Let $f(x) = \frac{2}{1+\exp(-(x^2 w_1 + x w_2 + w_3))} - 1$ and $L(y, f(x)) = \log(1+\exp(-yf(x)))$, with $y \in \{-1, 1\}$. Provide the gradient descent solution $\frac{\partial L}{\partial w_i}$ for $i = \{1, 2, 3\}$.**

   ....

The solution can be expressed in various ways. Here is a simple derivation in the spirit of artificial neural networks. Let

$$h(x) = \frac{2}{1+\exp(-x)} - 1 \tag{1}$$

and

$$g(x) = x^2 w_1 + x w_2 + w_3 \tag{2}$$

we have

$$\begin{aligned} f(x) &= \frac{2}{1+\exp(-(x^2 w_1 + x w_2 + w_3))} - 1 && (3) \\ &= \frac{2}{1+\exp(-g(x))} - 1 && (4) \\ &= h(g(x)) && (5) \\ & && (6) \end{aligned}$$

and then

$$\frac{\partial h(x)}{\partial x} = -\frac{h(x)^2 - 1}{2} \tag{7}$$

and

$$\begin{aligned} \frac{\partial g(x)}{\partial w_1} &= x^2 && (8) \\ \frac{\partial g(x)}{\partial w_2} &= x && (9) \\ \frac{\partial g(x)}{\partial w_3} &= 1 && (10) \\ & && (11) \end{aligned}$$

furthermore,

$$\begin{aligned} L(y, f(x)) &= \log(1+\exp(-yf(x)) && (12) \\ \frac{\partial L}{\partial f(x)} &= -\frac{y}{1+\exp(yf(x))} && (13) \end{aligned}$$

so

$$\begin{aligned} \frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial f(x)} \frac{\partial f(x)}{\partial h(x)} \frac{\partial h(x)}{\partial g(x)} \frac{\partial g(x)}{\partial w_1} && (14) \\ &= -\frac{y}{1+\exp(yf(x))} \cdot -\frac{h(g(x))^2 - 1}{2} \cdot x^2 && (15) \\ \frac{\partial L}{\partial w_2} &= -\frac{y}{1+\exp(yf(x))} \cdot -\frac{h(g(x))^2 - 1}{2} \cdot x && (16) \\ \frac{\partial L}{\partial w_3} &= -\frac{y}{1+\exp(yf(x))} \cdot -\frac{h(g(x))^2 - 1}{2} \cdot 1 && (17) \\ & && (18) \end{aligned}$$

. . . .

4. **(a) Provide the gradient descent solution for an MLP f with 2 layers, and a cost function $\mathcal{C}(y, f(x))$.**

   (b) Copying `mlp.py` implement an MLP with 2 layers.

   (c) Compare on a 2-dimensions dataset, the decision functions of an MLP with 1 layer and 2 layers. Take a look at the decision functions.

. . . .

The equation for an MLP f with 2 layers:

$$out = f(input) = v \cdot z_2 \left\{ y_2 \left[z_1 \left(y_1(input) \right) \right] \right\} + c$$

where,

- $input \in \mathbb{R}^n$, $out \in \mathbb{R}$,
- $y_1(input) = w_1 \cdot input + b_1 = (\sum_{l=1}^{n} w_1^{jl} input^l + b_1^j)_{j=1 \ldots nhu_1}$,
- $z_1 = (h(y_1^1), \ldots, h(y_1^{nhu_1}))$,
- $y_2(z_1) = w_2 \cdot z_1 + b_2 = (\sum_{j=1}^{nhu_1} w_2^{ij} z_1^j + b_2^i)_{i=1 \ldots nhu_2}$,
- $z_2 = (h(y_2^1), \ldots, h(y_2^{nhu_2}))^t$,
- h is a transfer function (*eg tanh*),
- w_1 is the $nhu_1 \times n$ 1st layer weight matrix (nhu_1: number of hidden units for the 1st layer),
- b_1 is the nhu_1 1st layer bias vector,
- w_2 is the $nhu_2 \times nhu_1$ 2nd layer weight matrix (nhu_2: number of hidden units for the 2nd layer),
- b_2 is a nhu_2 2nd layer bias vector,
- v is the $1 \times nhu_2$ output layer weight matrix and
- b is the output layer bias.

The gradients:

$$\frac{\partial f}{\partial v^i} = z_2^i, \quad \frac{\partial f}{\partial c} = 1, \quad \frac{\partial f}{\partial z_2^i} = v^i$$

$$\frac{\partial z_2}{\partial y_2} = \left(\frac{\partial h(y_2^1)}{\partial y_2^1}, \ldots, \frac{\partial h(y_2^{nhu_2})}{\partial y_2^{nhu_2}} \right)^t_{nhu_2 \times 1}$$

$$\frac{\partial y_2^i}{\partial w_2^{ij}} = z_1^j, \quad \frac{\partial y_2^i}{\partial b_2^i} = 1, \quad \frac{\partial y_2^i}{\partial z_1^j} = w_2^{ij}$$

$$\frac{\partial z_1}{\partial y_1} = \left(\frac{\partial h(y_1^1)}{\partial y_1^1}, \ldots, \frac{\partial h(y_1^{nhu_1})}{\partial y_1^{nhu_1}} \right)^t_{nhu_1 \times 1}$$

$$\frac{\partial y_1^j}{\partial w_1^{jl}} = input^l, \quad \frac{\partial y_1^j}{\partial b_1^j} = 1, \quad \frac{\partial y_1^j}{\partial input^l} = w_1^l$$

$$\frac{\partial \mathcal{C}}{\partial v^i} = \frac{\partial \mathcal{C}}{\partial f} \cdot \frac{\partial f}{\partial v^i}, \quad \frac{\partial \mathcal{C}}{\partial c} = \frac{\partial \mathcal{C}}{\partial f} \cdot \frac{\partial f}{\partial c}$$

$$\frac{\partial \mathcal{C}}{\partial y_2^i} = \frac{\partial \mathcal{C}}{\partial f} \cdot \frac{\partial f}{\partial z_2^i} \cdot \frac{\partial z_2^i}{\partial y_2^i}$$

$$\frac{\partial \mathcal{C}}{\partial w_2^{ij}} = \frac{\partial \mathcal{C}}{\partial y_2^i} \cdot \frac{\partial y_2^i}{\partial w_2^{ij}}, \quad \frac{\partial \mathcal{C}}{\partial b_2^i} = \frac{\partial \mathcal{C}}{\partial y_2^i} \cdot \frac{\partial y_2^i}{\partial b_2^i}$$

$$\frac{\partial \mathcal{C}}{\partial y_1^j} = \sum_{i=1}^{nhu_2} \frac{\partial \mathcal{C}}{\partial y_2^i} \cdot \frac{\partial y_2^i}{\partial z_1^j} \cdot \frac{\partial z_1^j}{\partial y_1^j}$$

$$\frac{\partial \mathcal{C}}{\partial w_1^{jl}} = \frac{\partial \mathcal{C}}{\partial y_1^j} \cdot \frac{\partial y_1^j}{\partial w_1^{jl}}, \quad \frac{\partial \mathcal{C}}{\partial b_1^j} = \frac{\partial \mathcal{C}}{\partial y_1^j} \cdot \frac{\partial y_1^j}{\partial b_1^j}$$

....