Machine Learning
Exercise 6

Marc Toussaint

Machine Learning & Robotics lab, U Stuttgart

Universitsstrae 38, 70569 Stuttgart, Germany

June 10, 2013

1 SVMs

a) Draw a small dataset $\{(x_i, y_i)\}$, $x_i \in \mathbb{R}^2$ with two different classes $y_i \in \{0, 1\}$ such that a 1-nearest neighbor (1-NN) classifier has a lower leave-one-out cross validation error than a SVM classifier.

b) Draw a small dataset $\{(x_i, y_i)\}$, $x_i \in \mathbb{R}^2$ with two different classes $y_i \in \{0, 1\}$ such that 1-NN classifier has a higher leave-one-out cross validation error than a SVM classifier.

c) Proof that the constrained optimization problem

$$\min_\beta \|\beta\|^2 + C \sum_{i=1}^n \xi_i \quad \text{s.t.} \quad y_i(x_i^\top \beta) \ge 1 - \xi_i \ , \quad \xi_i \ge 0$$

can be rewritten as the unconstrained optimization problem

$$\min_\beta \|\beta\|^2 + C \sum_{i=1}^n \max\{0,\ 1 - y_i(x_i^\top \beta)\} \ .$$

(Note that the $\max$ term is the hinge loss.)

d) Explain why the optimal model (optimal β) will "not depend" on data points for which $\xi_i = 0$. Here "not depend" is meant in the variational sense: roughly, the derivative of β w.r.t. these data points is zero.

2 Neural Networks

(As preparation for the next lecture.)

A sober view on neural networks (NNs) is that they are an interesting class of parameterized functions $y = f(x, w)$ that map some input $x \in \mathbb{R}^n$ to some output $y \in \mathbb{R}$ depending on parameters w.

We've introduced the *logistic sigmoid function* as

$$\sigma(z) = \frac{e^z}{1 + e^z} = \frac{1}{e^{-z} + 1} \ , \quad \sigma'(z) = \sigma(z)(1 - \sigma(z))$$

A 1-layer NN, with h_1 neurons in the hidden layer, is defined as

$$f(x, w) = w_1^\top \sigma(W_0 x) \ , \quad w_1 \in \mathbb{R}^{h_1}, W_0 \in \mathbb{R}^{h_1 \times d}$$

with parameters $w = (W_0, w_1)$, where $\sigma(z)$ is applied component-wise.

A 2-layer NN, with h_1, h_2 neurons in the hidden layers, is defined as

$$f(x, w) = w_2^\top \sigma(W_1 \sigma(W_0 x)) \ , \quad w_2 \in \mathbb{R}^{h_2}, W_1 \in \mathbb{R}^{h_2, h_1}, W_0 \in \mathbb{R}^{h_1 d}$$

with parameters $w = (W_0, W_1, w_2)$.

The weights of hidden neurons W_0, W_1 are usually trained using gradient descent. (The output weights w_1 or w_2 can be optimized analytically as for linear regression.)

a) Derive the gradient $\frac{\partial}{\partial W_1} f(x)$ for the 1-layer NN.

b) Derive the gradients $\frac{\partial}{\partial W_1} f(x)$ and $\frac{\partial}{\partial W_2} f(x)$ for the 2-layer NN.