CSE176 Introduction to Machine Learning — Homework set #3
Fall semester 2023 — Miguel Á. Carreira-Perpiñán

Exercise 1: linear classifier (10 points). Consider a binary linear classifier $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ with $\mathbf{w} = \binom{2}{-3}$ and $w_0 = 6$, where $\mathbf{x} \in \mathbb{R}^2$. Let class 1 be its positive side ($g(\mathbf{x}) > 0$) and class 2 its negative side ($g(\mathbf{x}) < 0$).

1. (4 points) Sketch the decision boundary in $\mathbb{R}^2$. Compute the points at which it intersects the coordinate axes. Indicate which is the positive side of the boundary (class 1).
2. (4 points) Compute the signed distance of the following points to the decision boundary: the origin; $\binom{-2}{2}$; $\binom{3}{4}$. Classify those points.
3. (2 points) Give a vector $\mathbf{u} \in \mathbb{R}^2$ that is parallel to the decision boundary and has norm 1.

Exercise 2: linear classifier (20 points). We have a classification problem with $K = 3$ classes in $\mathbb{R}^2$ with the following discriminant functions:

$$\begin{aligned} g_1(\mathbf{x}) &= -x_1 + 2x_2 + 4 \\ g_2(\mathbf{x}) &= x_1 + x_2 + 3 \\ g_3(\mathbf{x}) &= -2x_1 - x_2 + 1. \end{aligned}$$

1. (2 points) Give a rule to decide which class a point $\mathbf{x} \in \mathbb{R}^2$ should be assigned to.
2. (4 points) Classify the following points: $\binom{-1}{0}$, $\binom{2}{0}$, $\binom{-1}{-2}$, $\binom{1}{1}$.
3. (6 points) Give the equation that a point $\mathbf{x} \in \mathbb{R}^2$ must satisfy for it to be on the boundary between classes 1 and 2. Repeat for the boundary of class 1 and 3, and for class 2 and 3.
4. (2 points) Give the equation that a point $\mathbf{x} \in \mathbb{R}^2$ must satisfy for it to be on the boundary between all 3 classes.
5. (6 points) Based on the above, sketch the boundaries that delimit the 3 classes, indicating numerically where they cross the coordinate axes and which region corresponds to which class.

Exercise 3: one-vs-X classifiers (20 points). There are many types of binary classifiers, linear and nonlinear, and they have different training and test times. Assume we have access to a binary classifier model such that the time to train a classifier on M instances in dimension D is $\mathcal{O}(DM^\alpha)$, and the time to apply the trained binary classifier to a test instance is $\mathcal{O}(DM^\beta)$, where $\alpha, \beta \geq 0$. The table lists some examples:

Model	α	β
Logistic regression	1	0
Logistic regression with subsampling	$\frac{1}{2}$	0
Kernel SVM	2 or 3	1
Gaussian process	0	3

We want to construct a K-class classifier using binary classifiers with either the one-vs-all or the one-vs-one approach, on a training set containing N instances in dimension D, where the classes are balanced (i.e., each class contains N/K instances). For each of the two approaches, and assuming the general formulas above (as a function of $\alpha, \beta \geq 0$), compute (using asymptotic big-O notation, and explaining how you derive your result):

1. (5 points) The training time on the N instances. When is it better to use one-vs-all rather than one-vs-one?
2. (5 points) The testing time on one instance. When is it better to use one-vs-all rather than one-vs-one?

Then:

3. (8 points) Complete the table above with 4 additional columns, each giving the asymptotic training and test time for the one-vs-all and one-vs-all approaches for each of the 4 models in the table. Is one-vs-all the better approach?
4. (2 points) Consider the following widely used implementations:
   - LIBLINEAR: K-class linear classifier using logistic regression as base classifier with the one-vs-all approach.
   - LIBSVM: K-class nonlinear classifier using a kernel SVM as base classifier with the one-vs-one approach.

   Are the LIBLINEAR and LIBSVM choices justified?

Exercise 4: logistic regression (27 points). Consider a binary classification problem in dimension D with a training set $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where $\mathbf{x}_n \in \mathbb{R}^D$ and $y_n \in \{0,1\}$ for $n = 1, \dots, N$.

1. (2 points) Write the cross-entropy objective function $E(\mathbf{w}, w_0)$ for logistic regression.
2. (4 points) Compute and simplify the gradient of E with respect to the parameters $\mathbf{w} \in \mathbb{R}^D$ and $w_0 \in \mathbb{R}$.
3. (2 points) Write the update formulas for the parameters using gradient descent with a step size $\eta > 0$.
4. (4 points) Consider a training set in dimension 1 that is symmetric around the origin: it contains $\frac{N}{2}$ points at positive locations $x_1, \dots, x_{N/2} > 0$ with class label 1, and $\frac{N}{2}$ points at negative locations $-x_1, \dots, -x_{N/2} < 0$ with class label 0. Using intuition, guess the optimal parameter values (w^*, w_0^*).
5. (10 points) Prove that (w^*, w_0^*) in the previous question is in fact a minimizer of the cross-entropy, by verifying the following: 1) that the gradient of E is zero at (w^*, w_0^*); and 2) that (w^*, w_0^*) is a global minimizer, by showing that no other $w, w_0 \in \mathbb{R}$ can achieve a lower objective function value.
6. (5 points) The minimizer (w^*, w_0^*) above is peculiar. Why? Explain: would this situation happen in dimension $D > 1$? What happens with the class posterior probabilities, compared to a typical logistic regression? What is the fundamental reason for this peculiar situation? How would you prevent it?

In all of the above, show your work and explain the result. Just giving the final result earns no points, even if correct. Useful facts about the logistic function: derivative $\frac{d\sigma(x)}{dx} = \sigma'(x) = \sigma(x)(1 - \sigma(x))$; and $\sigma(x) + \sigma(-x) = 1$.

Exercise 5: multilayer perceptrons (8 points). Consider the following training set of 2D points in two classes $\{0,1\}$:

$$\{(\mathbf{x}_n, y_n)\}_{n=1}^N = \left\{ \left(\binom{0}{0}, 0\right), \left(\binom{0}{1}, 0\right), \left(\binom{1}{0}, 0\right), \left(\binom{1}{1}, 0\right), \left(\binom{1/2}{1/2}, 1\right) \right\}.$$

Construct manually a multilayer perceptron that classifies it perfectly (giving numerical values of the MLP's parameters). The smaller your MLP (number of layers and units) the better the grade.

Exercise 6: gradient of multilayer networks (15 points). Consider the following function $f\colon \mathbb{R}^D \to (0,1)$ that maps input vectors $\mathbf{x} \in \mathbb{R}^D$ to output values $y \in (0,1)$ (for use in binary classification):

$$f(\mathbf{x}) = \sigma\left(\sum_{h=1}^H \alpha_h e^{-\frac{1}{2}\left\| \frac{\mathbf{x} - \boldsymbol{\mu}_h}{\sigma} \right\|^2} \right)$$

where we use the logistic function $\sigma(x) = \frac{1}{1+e^{-x}} \in (0,1)$ for $x \in \mathbb{R}$. The parameters of f are the weights $\{\alpha_h\}_{h=1}^H \subset \mathbb{R}$, centers $\{\boldsymbol{\mu}_h\}_{h=1}^H \subset \mathbb{R}^D$ and bandwidth $\sigma > 0$. The function f can be seen as either a neural net with a single hidden layer of Gaussian units, or a logistic regression whose input is a Gaussian radial basis function. Consider a training set for binary classification $\{(\mathbf{x}_n, y_n)\}_{n=1}^N \in \mathbb{R}^D \times \{0,1\}$. We want to train f by minimizing the squared error as loss function:

$$E\left(\{\alpha_h, \boldsymbol{\mu}_h\}_{h=1}^H, \sigma\right) = \sum_{n=1}^N (y_n - f(\mathbf{x}_n))^2 + \lambda \sum_{h=1}^H \alpha_h^2$$

where $\lambda \geq 0$ is a fixed regularization hyperparameter.

1. (6 points) Compute the gradients of E wrt the parameters of f, simplifying them as much as possible. Useful facts about the logistic function: derivative $\frac{d\sigma(x)}{dx} = \sigma'(x) = \sigma(x)(1 - \sigma(x))$; and $\sigma(x) + \sigma(-x) = 1$.
2. (3 points) What would be a good initialization for these parameters (to start gradient descent)?

(6 points) Repeat all of the above but considering now the following function $f\colon \mathbb{R}^D \to (0,1)$:

$$f(\mathbf{x}) = \sigma\left(\sum_{h=1}^H \alpha_h \tanh\left(\boldsymbol{\beta}_h^T \mathbf{x} + \beta_{h0}\right) \right)$$

where we use the hyperbolic tangent function $\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \in (-1,1)$ for $x \in \mathbb{R}$. The function f can be seen as a neural net with a single hidden layer of tanh units.