ETC3555 2018 - Lab 4

Linear models and gradient descent

Cameron Roach and Souhaib Ben Taieb

15 August 2018

Exercise 1

Solve exercise 3.7 in Learning From Data.

Exercise 2

Solve exercise 3.9 in Learning From Data.

Exercise 3

Solve problem 3.16 in Learning From Data.

Exercise 4

Solve problem 3.4 in Learning From Data.

Exercise 5

In this exercise, we will implement (stochastic) gradient descent for linear regression and logistic regression. You will try your algorithm using various learning rates and stopping criteria (e.g. number of iterations). Make sure you pick a learning rate within a good range. For linear regression, compare your solution with the closed-form solution. Finally, we will also try L_2 regularization for both linear and logistic regression.

Exercise 3.7

For logistic regression, show that

$$\begin{aligned}\nabla E_{\text{in}}(\mathbf{w}) &= -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n\mathbf{x}_n}{1+e^{y_n\mathbf{w}^{\mathrm{T}}\mathbf{x}_n}} \\ &= \frac{1}{N}\sum_{n=1}^{N}-y_n\mathbf{x}_n\theta(-y_n\mathbf{w}^{\mathrm{T}}\mathbf{x}_n).\end{aligned}$$

Argue that a 'misclassified' example contributes more to the gradient than a correctly classified one.

Figure 1: Source: Abu-Mostafa et al. Learning from data. AMLbook.

Exercise 3.9

Consider pointwise error measures $\mathsf{e}_{\text{class}}(s,y) = [\![y \neq \text{sign}(s)]\!]$, $\mathsf{e}_{\text{sq}}(s,y) = (y-s)^2$, and $\mathsf{e}_{\text{log}}(s,y) = \ln(1+\exp(-ys))$, where the signal $s = \mathbf{w}^{\mathrm{T}}\mathbf{x}$.

(a) For $y = +1$, plot $\mathsf{e}_{\text{class}}$, e_{sq} and $\frac{1}{\ln 2}\mathsf{e}_{\text{log}}$ versus s, on the same plot.

(b) Show that $\mathsf{e}_{\text{class}}(s,y) \leq \mathsf{e}_{\text{sq}}(s,y)$, and hence that the classification error is upper bounded by the squared error.

(c) Show that $\mathsf{e}_{\text{class}}(s,y) \leq \frac{1}{\ln 2}\mathsf{e}_{\text{log}}(s,y)$, and, as in part (b), get an upper bound (up to a constant factor) using the logistic regression error.

These bounds indicate that minimizing the squared or logistic regression error should also decrease the classification error, which justifies using the weights returned by linear or logistic regression as approximations for classification.

Figure 2: Source: Abu-Mostafa et al. Learning from data. AMLbook.

Problem 3.16 In Example 3.4, it is mentioned that the output of the final hypothesis $g(\mathbf{x})$ learned using logistic regression can be thresholded to get a 'hard' (± 1) classification. This problem shows how to use the risk matrix introduced in Example 1.1 to obtain such a threshold.

Consider fingerprint verification, as in Example 1.1. After learning from the data using logistic regression, you produce the final hypothesis

$$g(\mathbf{x}) = \mathbb{P}[y = +1 \mid \mathbf{x}],$$

which is your estimate of the probability that $y = +1$. Suppose that the cost matrix is given by

		True classification	
		$+1$ (correct person)	-1 (intruder)
you say	$+1$	0	c_a
	-1	c_r	0

For a new person with fingerprint $\mathbf{x}$, you compute $g(\mathbf{x})$ and you now need to decide whether to accept or reject the person (i.e., you need a hard classification). So, you will accept if $g(\mathbf{x}) \geq \kappa$, where κ is the threshold.

(a) Define the cost(accept) as your expected cost if you accept the person. Similarly define cost(reject). Show that

$$\begin{aligned} \text{cost(accept)} &= (1 - g(\mathbf{x}))c_a, \\ \text{cost(reject)} &= g(\mathbf{x})c_r. \end{aligned}$$

(b) Use part (a) to derive a condition on $g(\mathbf{x})$ for accepting the person and hence show that

$$\kappa = \frac{c_a}{c_a + c_r}.$$

(c) Use the cost-matrices for the Supermarket and CIA applications in Example 3.4 to compute the threshold κ for each of these two cases. Give some intuition for the thresholds you get.

Figure 3: Source: Abu-Mostafa et al. Learning from data. AMLbook.

Problem 3.4 In Problem 1.5, we introduced the Adaptive Linear Neuron (Adaline) algorithm for classification. Here, we derive Adaline from an optimization perspective.

(a) Consider $E_n(\mathbf{w}) = \max(0, 1 - y_n \mathbf{w}^{\mathsf{T}} \mathbf{x}_n)^2$. Show that $E_n(\mathbf{w})$ is continuous and differentiable. Write down the gradient $\nabla E_n(\mathbf{w})$.

(b) Show that $E_n(\mathbf{w})$ is an upper bound for $[\![\text{sign}(\mathbf{w}^{\mathsf{T}} \mathbf{x}_n) \neq y_n]\!]$. Hence, $\frac{1}{N} \sum_{n=1}^{N} E_n(\mathbf{w})$ is an upper bound for the in-sample classification error $E_{\text{in}}(\mathbf{w})$.

(c) Argue that the Adaline algorithm in Problem 1.5 performs stochastic gradient descent on $\frac{1}{N} \sum_{n=1}^{N} E_n(\mathbf{w})$.

Figure 4: Source: Abu-Mostafa et al. Learning from data. AMLbook.

Problem 1.5 We know that the perceptron learning algorithm works like this: In each iteration, pick a random $(\mathbf{x}(t), y(t))$ and compute $\rho(t) = \mathbf{w}^{\mathrm{T}}(t)\mathbf{x}(t)$. If $y(t) \cdot \rho(t) \leq 0$, update $\mathbf{w}$ by

$$\mathbf{w}(t+1) \longleftarrow \mathbf{w}(t) + y(t) \cdot \mathbf{x}(t) \ ;$$

One may argue that this algorithm does not take the 'closeness' between $\rho(t)$ and $y(t)$ into consideration. Let's look at another perceptron learning algorithm: In each iteration, pick a random $(\mathbf{x}(t), y(t))$ and compute $\rho(t)$. If $y(t) \cdot \rho(t) \leq 1$, update $\mathbf{w}$ by

$$\mathbf{w}(t+1) \longleftarrow \mathbf{w}(t) + \eta \cdot (y(t) - \rho(t)) \cdot \mathbf{x}(t) \ ,$$

where η is some constant. That is, if $\rho(t)$ agrees with $y(t)$ well (their product is > 1), the algorithm does nothing. On the other hand, if $\rho(t)$ is further from $y(t)$, the algorithm changes $\mathbf{w}(t)$ more. In this problem, you are asked to implement this algorithm and check its performance.

(a) Generate a training data set of size 100 similar to that used in Exercise 1.4. Generate a test data set of size $10,000$ from the same process. To get g, run the algorithm above with $\eta = 100$ on the training data set, until it converges (no more possible updates) or a maximum of $1,000$ updates has been reached. Plot the training data set, the target function f, and the final hypothesis g on the same figure. Report the error on the test set.

(b) Use the data set in (a) and redo everything with $\eta = 1$.

(c) Use the data set in (a) and redo everything with $\eta = 0.01$.

(d) Use the data set in (a) and redo everything with $\eta = 0.0001$.

(e) Compare the results that you get from (a) to (d).

The algorithm above is a variant of the so-called Adaline (*Adaptive Linear Neuron*) algorithm for perceptron learning.

Figure 5: Source: Abu-Mostafa et al. Learning from data. AMLbook.