Optmization Methods for Machine Learning - Fall 2016

Assignment # 1 Perceptron

Laura Palagi

Department of Computer, Control, and Management Engineering Antonio Ruberti

Sapienza Università di Roma

Posted on October 4, 2016 - due date October 24, 2016

Instructions

Homework will be done individually: each student must hand in their own answers. It is acceptable for students to collaborate in figuring out answers and helping each other solve the problems. We will be assuming that, as participants in a graduate course, you will be taking the responsibility to make sure you personally understand the solution to any work arising from such collaboration.

Homework must be send by an email both to the teaching assistant Ing. Umberto Dellepiane (umberto.dellepiane@act-OperationsResearch.com) and to laura.palagi@uniroma1.it with subject **[OMML-2016] Project 1**. After you submit, you will receive an acknowledgement email that your project has been received. If you have not received an acknowledgement email within 2 days after you submit then contact the instructor.

The mail must contain as attachment a .zip or .tar.gz file both a typed report in English and the source code (including instructions to run it).

Evaluation criteria

The first homework accounts for 20% of the total vote of the exam.
Homework is due at latest at midnight on the due date. For late homework, the score will be decreased. It is worth 85% for the next 48 hours. It is worth 70% from 48 to 120 hours after the due date. It is worth 50% credit after 120 hours delay.
For the evaluation of the first homework the following criteria will be used:

1. check of the implementation (40% Umberto Dellepiane)
2. Quality of the explanation document and of the overall job (60% - Laura Palagi).

The grade are Italian style namely in the range [0,30], being 18 the minimum degree to pass the exam.

Perceptron Learning

In this assignment you will implement the perceptron algorithm and its variant for multiclass classification.

The picture represents a set of two-dimensional input samples from two classes linearly separable. The pairs (x^i, y^i) with $x^i \in \mathbb{R}^2$ and $y^i \in \{-1, 1\}$. The coordinates x^i can be estimated from the chart. A single perceptron should be able to learn this classification task perfectly by identifying $w = (w_1,\ w_2)^T$, $b \in \mathbb{R}$ such that $f(x) = g(w^T x + b)$ is the classification function where the activation function

$$g(t) := \mathrm{sgn}(t) = \begin{cases} 1 & t \geq 0 \\ -1 & t < 0. \end{cases}$$

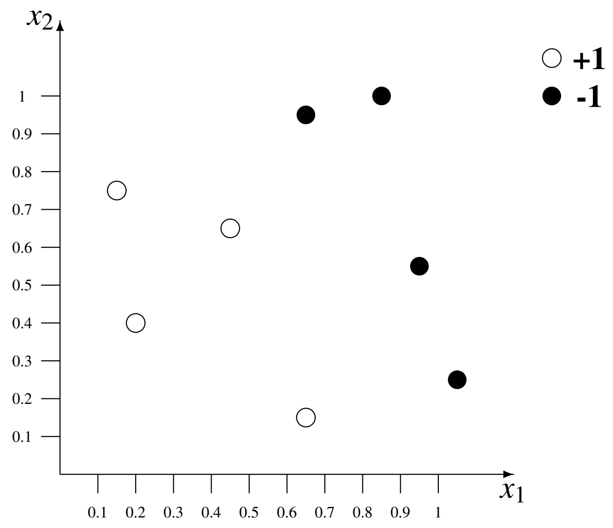

Figure 1: Sample in the two class

Question 1. (max score up to 24 (italian scale))

1. Report the values of the points defining the training set
2. Start with a perceptron with weights $b^0 = 0.2, w_1^0 = 1$, and $w_2^0 = -1$. Add the perceptrons initial line of division to the chart. How many samples are misclassified?
3. Pick an arbitrary misclassified sample and describe the computation of the weight update (the full first iteration of the perceptron). Plot the perceptrons new line of division in the same chart or a different one, and give the new number of misclassified samples.
4. Repeat this process four more times so that you have a total of six lines (or fewer if your perceptron achieves perfect classification earlier).

   You can do the computations above and and/or graphs either by hand or by writing a computer program (you can choose the language: matlab, phyton, etc.).

Question 2. (max score up to 30 (italian scale))

4. Write a program (please attach the code) that implements the simple perceptron on the data set above and let the program run until the perceptron achieves perfect classification. How many steps are needed ? ie
5. Modify the code to implement the so called *Averaged Perceptron*. It consists in maintaining a weight vector w^{avg} that is the average of all the weight vectors after each iteration. After training, return this weight vector instead of the final weight vector. The averaged perceptron is a modification of the voting perceptron. Voting algorithm remembers how long each hyperplane survives. For example if an hyperplane survived for 10 examples, then it gets a vote of 10. If it only survived for one example, it only gets a vote of 1. Let $(w,b)^k$ for $k = 1,\dots,P$ vectors encountered in the algorithm and $v^1,\dots,v^P$ the corresponding survival time, then

$$w^{avg} = \sum_{k=1}^{P} v^k w^k \qquad b^{avg} = \sum_{k=1}^{P} v^k b^k$$

```
**Data.** Input $x^i$, with $\|x^i\| \le R$, Target $y^i$, $i = 1,\dots,\ell$.
**Inizialization.** Set $w^0 = 0, b^0 = 0$, $w^{avg} = 0, b^{avg} = 0$, $k = 0$, $v^k = 0$, Its=0.
**While** Its$\le$ Maxiter **do**
    **For** $i = 1,\dots,\ell$ **do**
            **If** $y^i \cdot sgn(w^{k^T} x^i + b^k) < 0$ **then**
              $w^{k+1} \leftarrow w^k + y^p x^p$ and
              $b^{k+1} \leftarrow b^k + y^p$
              $w^{avg} \leftarrow w^{avg} + v^k w^k$ and
              $b^{avg} \leftarrow b^{avg} + v^k b^k$
              $v^k = 1$ and $k = k+1$
            **else** $v^k = v^k + 1$,
    **End For**
Its=Its+1
**End While**
( do the last update of the $w^{avg}$ and $b^{avg}$ weights)
$w^{avg} \leftarrow w^{avg} + v^k w^k$ and $b^{avg} \leftarrow b^{avg} + v^k b^k$
**Return** $w^{avg}, b^{avg}$
```

Classifier take the form

$$f_{w^{avg},b^{avg}}(x) = sign(x^T w^{avg} + b^{avg})$$

In the report specify the value of ”Maxiter” that you have used and answer the following questions:

(a) How many iterations the algorithm performs to get a separating hyperplane ?

(b) How the algorithm performs varying the value of ”Maxiter” ?

6. Change one or more label of the original data set so that points are non more linearly separable, apply your average perceptron to this new set. What does it happen ?

Question 3. (max score up to 30 cum laude (italian scale))

7. Assume to use a different activation function g such as the *hyperbolic tangent*

$$g(t) := \tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}},$$

and let $f(x) = g(w^T x + b)$ be the classification function.

Write a program (please attach a printout) which implements the average quadratic error

$$E(w) = \frac{1}{2} \sum_{p=1}^{P} (y^p - f(x^p))^2 ,$$

and use a matlab routine of the optimization toolbox for its minimization, namely solve the problem $\min_{w,b} E(w,b)$. Please observe that we are not asking to write/evaluate the gradient of $E(w)$. You can use a matlab toolbox to avoid this calculus.