Übung zur Vorlesung

Maschinelles Lernen 2

Sommersemester 2009

Abteilung Maschinelles Lernen
Institut für Softwaretechnik und
theoretische Informatik
Fakultät IV, Technische Universität Berlin
Prof. Dr. Klaus-Robert Müller
Email: krm@cs.tu-berlin.de

Blatt 11

Abgabe bis *Mittwoch*, 13. Juli 2009 bis 13 Uhr, Ausarbeitung im Sekretariat FR6052, oder bei Mikio Braun, FR6058, Lösung des praktischen Teils per Email an mikio@cs.tu-berlin.de.

Aufgaben

In der Vorlesung wurden Large-Scale-Lernmethoden anhand von Support Vector Machines besprochen. In der Übung soll die der stochastische Gradienten Abstieg weiter vertieft werden, da er besonders einfach zu implementieren und in der Praxis erstaunlich gute Resultate liefert.

1. Zeige, dass das primale SVM Problem mit Nebenbedingungen

$$\begin{aligned} \min_{\boldsymbol{w}\in\boldsymbol{R}^D, b\in\boldsymbol{R}, \xi\in\boldsymbol{R}^N} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^N \xi_i \\ \text{so dass} \quad & \xi_i \geq 0, && 1 \leq i \leq n \\ & y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i, && 1 \leq i \leq n \end{aligned}$$

äquivalent zu

$$\frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^N \left(\max\{0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\}\right)$$

ist. **(8 Punkte)**

2. Leite die Update-Regeln für Stochastic Gradient Descent für quadratische Verlustfunktionen

$$\frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^N \left(\max\{0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i)\}\right)^2$$

her. **(8 Punkte)**

3. Implementiere Stochastic Gradient Descent für die Verlustfunktion in Aufgabe 2 im Programmskelett `sheet11.m`. Die Optimierung soll aus einer gegebenen Anzahl von Epochen bestehen, in der das Gewicht jeweils wieder auf den Startwert gesetzt werden soll. Während einer Epoche soll das Gewicht wie 1/Anzahl der Iterationen abfallen.

   D.h. die grobe Struktur ist wie folgt:

```
draw random permutation p over data points
for e in 1:epochs
  for i in 1:iterations
    if p is empty
      draw new random permutation p
    get the next point x, y using p and remove it from the permutation
    compute gradient for point x, y
    subtract gradient from w with weight eta/i
  end
  plot training and test error
end
```

   (14 Punkte)

```
function sheet11

fprintf('loading data...');
X = load('alpha_train_x.txt');
Y = load('alpha_train_y.txt');

[N, D] = size(X);

fprintf('%d data points with %d features.\n', N, D);

P = randperm(N);

XE = X(P(N/2:end), :);
YE = Y(P(N/2:end), :);
X = X(P(1:N/2), :);
Y = Y(P(1:N/2), :);

size(X)
size(Y)
size(XE)
size(YE)

fprintf('training...');
% find reasonable values below!
epochs = ?;
iters = ?;
C = ?;
eta = ?;
C = linsvm_sgd(C, eta, epochs, iters, X, Y, XE, YE);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% your solution below.
%
%

% Train a linear SVM with quadratic costs on the slack variables using
% stochastic gradient descent.
%
% Parameters:
%   - eta: base value for eta
%   - epochs: total number of runs through the data.
%   - iters: number of iterations per epoch
%   - X, Y: training data
%   - XE, YE: test data (only for computing the progress)
%
% Plot the test errors on training and test data for each epoch.
function linsvm_sgd(C, eta, epochs, iters, X, Y, XE, YE)
% ...
```