Übung zur Vorlesung

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

Maschinelles Lernen 1

Wintersemester 2008/2009

Blatt 5

Abgabe bis Mittwoch, 19. November 2008, 14:00 Uhr
Ausarbeitung im Sekretariat FR6052, oder bei Mikio Braun, FR6058 (notfalls unter der Tür durchschieben),
praktischen Teil unter https://ml01.service.tu-berlin.de/˜mikio/pass.pl?conf=blatt5.conf.

Aufgaben

Weiterhin ist darauf zu achten, so wenig Schleifen wir möglich zu verwenden und statt dessen Matrix und Vektoroperationen zu verwenden.

1. **Bayesianische Inferenz und Maximum-Likelihood-Schätzung** In dieser Aufgabe sollen die Bayesianische Inferenz und die ML Schätzung auf dem einfachen Beispiel einer Bernoullivariable verglichen werden.

   (a) Wir nehmen als Priorverteilung die Betaverteilung an. Dies ist eine Dichte auf dem Intervall $[0, 1]$ mit der Dichtefunktion

   $$B(p|a, b) = \frac{\Gamma(a + b)}{\Gamma(a)\Gamma(b)} p^{a-1}(1 - p)^{b-1}.$$

   Zeige, dass bei der Beobachtung einer einzigen Realisation $X \in \{0, 1\}$, die Posteriorverteilung wieder Betaverteilt ist mit

   $$p \sim \begin{cases} B(p|a + 1, b) & \text{falls } X = 1 \\ B(p|a, b + 1) & \text{falls } X = 0. \end{cases}$$

   (Hinweis: Vernachlässige zunächst die Normierung und zeige, dass man durch Anwenden der Bayesformel wieder eine Betaverteilung erhält (bis auf Normierung). Setze dann wieder die korrekte Normierung ein.)

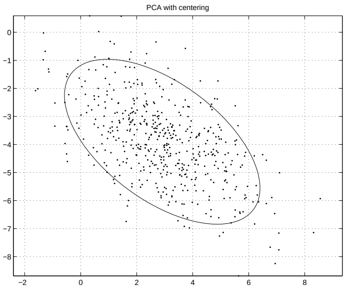
   (b) Die beiden Schätzer sollen praktisch verglichen werden. Ergänze hierfür die fehlnden Funktionen im Programmskelett `sheet05_01.m`:

   - `ml_estimate` Berechne für einen Vektor X mit Werten 0 und 1 den ML-Schätzer für den Mittelwert der Bernoulliverteilung.
   - `beta_update` Berechne einen Schritt in der Bayesschen Inferenz wenn man eine Realisierung beobachtet hat.
   - `beta_stats` Bestimme für Vektoren A und B den Mittelwert und die Varianz der Betaverteilung. Die Formeln lauten

   $$\text{Mittelwert: } \frac{a}{a + b} \qquad \text{Varianz: } \frac{ab}{(a + b)^2(a + b + 1)}.$$

2. **PCA** Implementiere die Hauptkomponentenanalyse (PCA) und eine Funktion, die die Hauptkomponenten als Ellipse darstellt. Verwende hierzu das Programmskelett `sheet05_02.m` und ergänze die folgenden Funktionen:

   - `pca` soll die PCA berechnen, wahlweise ohne vorherige Zentrierung. Das Ergebnis soll als Struktur zurückgegeben werden mit den Feldern U (Matrix Hauptkomponentenrichtungen als Spalten), D (Vektor mit Hauptkomponentenwerten), M (Vektor mit dem Mittelpunkt der Daten).
   - `plot_pca` soll die Daten und die PCA darstellen. Hierfür soll eine entsprechend gedrehte Ellipse gemalt werden, deren Radii $2\sqrt{d}$ entspricht, wobei d die Hauptkomponentenwerte sind. Als Beispiel:



```
function sheet05_01
% Comparing Bayesian estimation versus maximum-likelihood estimation.

% generate some Bernoulli distributed data
N = 300;
p = 0.3;
X = rand(1, N) < p;

% estimate p using the ml estimator
ml = zeros(1, N);
for I = 1:N
  ml(I) = ml_estimate(X(1:I));
end

% estimate p using a "non-informative"
[B11A, B11B] = beta_estimate(1, 1, X);
[B7030A, B7030B] = beta_estimate(70, 30, X);
[B300700A, B300700B] = beta_estimate(300, 700, X);

B11A

plot(1:N, ml, 'g');
hold on;
plot_beta_estimates(B11A, B11B, 'r');
plot_beta_estimates(B7030A, B7030B, 'k');
plot_beta_estimates(B300700A, B300700B, 'b');
hold off;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% only make changes blow

% 1. Compute the maximum-likelihood estimate for Bernoulli random
% variables from the samples contained in X
function p = ml_estimate(X)
% ...

% Compute the continuous updates of the beta distribution prior
% parameters for X.
function [A, B] = beta_estimate(a, b, X)
N = length(X);
A = zeros(1, N);
```

```
B = zeros(1, N);
for I = 1:N
  [a, b] = beta_update(a, b, X(I));
  A(I) = a;
  B(I) = b;
end

% 2. Update the a and b parameters of the beta distribution prior
% for a single X.
function [a, b] = beta_update(a, b, X)
% ...

% 3. Compute the mean and variance of the beta distribution for
% the given vector of parameters
function [M, V] = beta_stats(A, B)
% ...

% Plot mean and two times standard deviation.
function plot_beta_estimates(A, B, mark)
N = length(A);
[M, V] = beta_stats(A, B);
plot(1:N, M, [mark, '-'], ...
     1:N, M + 2*sqrt(V), [mark, ':'], ...
     1:N, M - 2*sqrt(V), [mark, ':']);
```

```
function sheet05_02
% Principal Component Analysis

% simple data set
X = randn(500, 2) * diag([1, 2]) * rotmat(0.3*pi);
X = X + repmat([3, -4], 500, 1);

plot(X(:,1), X(:,2), '.');
axis equal
grid

% first PCA without centering
figure(1)
p = pca(X, 0)
plot_pca(X, p);
title('PCA without centering')

% now, with proper centering
figure(2)
p = pca(X)
plot_pca(X, p);
title('PCA with centering')

% and now, we add further, far away point
figure(3)
X = [X; -50, -50];
p = pca(X);
plot_pca(X, p);
title('PCA is not very stable with respect to outliers')

% how about other data?
figure(4)
phi = rand(500, 1) * 4 * pi;
X = [phi, cos(phi) + (phi.^3)/1e2] + 0.5*randn(500, 2);

p = pca(X);
```

```
plot_pca(X, p);

function R = rotmat(phi)
R = [cos(phi), sin(phi); -sin(phi), cos(phi)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Fill in your solutions below

% 1. Compute the principal components. If the second argument is given and
% non-zero, properly center the data. Otherwise, don't center.  Return a
% struct with fields "U", "D", and "M" with
%
%    U is the matrix whose columns are the principal components
%    D is a vector with the variances
%    M is the mean vector
function result = pca(X, center)
if nargin == 1
  center = 1;
end
% ...
result.U = U;
result.D = D;
result.M = M;

% 2. Plot the principal components for the PCA. First, plot the data as
% points. Then, draw an ellipsis whose radii are given as 2*sqrt(D) such
% that the ellipsis contains the data more or less. Add a grid and make sure
% axes are sized equally.
function plot_pca(X, p)
% ...
```