CHALMERS UNIVERSITY OF TECHNOLOGY

# Homework 0

TDA231 - Algorithms for Machine learning & Inference

Fager Jonathan fagerj@student.chalmers.se 920212-0052

Lindqvist Jakob jaklindq@student.chalmers.se 921028-5574

January 23, 2017

# 1 Theoretical problems

## 1.1 Bayes' rule

Calculating the conditional probability of being sick given that the test returns positive using Bayes' rule:

$$P(\text{sick} \mid \text{positive}) = \frac{P(\text{positive} \mid \text{sick})P(\text{sick})}{P(\text{positive})}, \tag{1}$$

where the quantities in the nominator are given in the task (0.99 and 0.0001 respectively). The denominator remains to be calculated. The probability of a test returning positive regardless of your actual health can be calculated as the joint probabilities of

$$\begin{aligned} P(\text{positive}) &= P(\text{sick })P(\text{test correct}) + P(\text{healthy })P(\text{test wrong }) = \\ &= 0.0001 \cdot 0.99 + 0.9999 \cdot 0.01 = 0.0101 \end{aligned} \tag{2}$$

Using this in equation (1) we acquire the final result

$$P(\text{sick} \mid \text{positive }) = \frac{0.99 \cdot 0.0001}{0.0101} = 0.0098, \tag{3}$$

## 1.2 Covariance

A useful identity for the covariance between two jointly distributed random variables is

$$\text{cov}(X, Y) = E(XY) - E(X)E(Y). \tag{4}$$

The expected value of $X$ is obviously zero since it is uniformly distributed over a symmetric interval, and the expected value of $Y := X^2$ is at least bounded and the last term vanishes and we only need to calculate the third moment of $X$:

$$\text{cov}(X, Y) = E(XY) = E(X^3) = \int_{-\infty}^{\infty} x^3 f(x) dx = \frac{1}{2} \int_{-1}^{1} x^3 dx, \tag{5}$$

where $f(x)$ is the pdf of $X$. This is an integral of an odd function over a symmetric interval and the covariance is indeed zero.

# 2 Practical problems

## 2.1 Plotting normal distribution

We draw a 1000 points from the bivariate normal distribution that is predefined in matlab as `mvnrnd`. With parameters

$$\Sigma = \begin{bmatrix} 0.1 & -0.05 \\ -0.05 & 0.02 \end{bmatrix}, \mu = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{6}$$

we define the function

$$f(\vec{x}, r) = \frac{(\vec{x} - \vec{\mu})^T \Sigma^{-1} (\vec{x} - \vec{\mu})}{2} - r. \tag{7}$$

The level curves of the function were calculated and plotted using `ezplot` which takes a function as an argument and generates a plot showing the curve for which the function is zero. This was done using the different values for the parameter, $r = \{1, 2, 3\}$.
$f(\vec{x}, r)$ is a quadratic form described by the matrix $\Sigma^{-1}$ which is positive definite which in turn means that every stationary point is a minimum. It is easy to realise that $f$ only has one minimum and that it is obtained at $\vec{x} = \vec{\mu} = (1, 1)^T$, furthermore $f(\vec{\mu}, r) < 0, \forall r > 0$. This means that finding every point "outside" the level curve $f(\vec{x}, 3) = 0$ is equivalent to finding all $x$ for which $f(x, 3) > 0$ is true. We achieve this by simply evalutating the samples drawn from the distribution and separate the data using this criterion. The result is shown together with the level sets for different values of $r$ in figure (1)

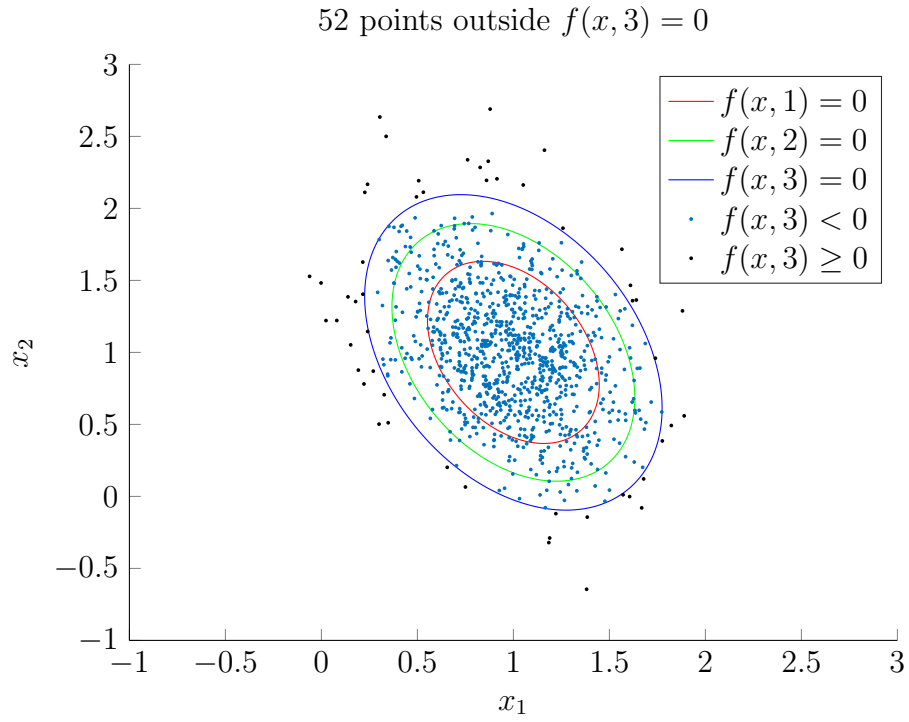


Figure 1: Scatter plot of the $n = 1000$ point sample generated from the bivariate normal distribution with parameters $\vec{\mu} = (1, 1)^T$ and $\Sigma = \begin{bmatrix} 0.1 & -0.05 \\ -0.05 & 0.02 \end{bmatrix}$. The points inside and outside of the level set $f(\vec{x}, 3) = 0$ are separated by colour. The level sets for $r = \{1, 2, 3\}$ are also plotted.

## 2.2 Covariance and Correlation

Scaling the dataset $X$ to take values between zero and one

$$Y = \frac{X - \min(X)}{\max(X) - \min(X)} := \frac{X - X_{\min}}{\overline{X}} \tag{8}$$

Using this transformation we can calculate the correlation between $X$ and $Y$. The correlation is defined as

$$\rho_{XY} = \operatorname{corr}(X, Y) = \frac{\operatorname{cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}. \tag{9}$$

Now consider

$$Y - \mu_Y = \frac{X - X_{\min}}{\overline{X}} - E\left(\frac{X - X_{\min}}{\overline{X}}\right); \tag{10}$$

using the linearity of expectation and the fact that the minimum and maximum are deterministic we obtain

$$\frac{X - X_{\min}}{\overline{X}} - \frac{1}{\overline{X}}\left(\mu_X - X_{\min}\right) = \frac{1}{\overline{X}}\left(X - \mu_X\right) \tag{11}$$

Using the same argument as above we express the standard devation of $Y$ in terms of $X$

$$\sigma_Y := \sqrt{E[(Y-\mu_Y)^2]} = \sqrt{E[\frac{1}{\overline{X}^2}(X-\mu_X)^2]} = \frac{1}{\overline{X}}\sqrt{E[(X-\mu_X)^2]} \tag{12}$$

Inserting this in equation (9) renders

$$\mathrm{corr}(X,Y) = \frac{\frac{1}{\overline{X}}E[(X-\mu_X)(X-\mu_X)]}{\frac{1}{\overline{X}}\sigma_X\sigma_X} = \mathrm{corr}(X,X). \tag{13}$$

Hence we do not expect any changes in the correlation due to this transformation which we also can observe in figure (2); each feature in $X$ correlates perfectly with its transformed counterpart in $Y$.
A similar analysis can be performed for the covariance:

$$\begin{aligned}\mathrm{cov}(X,Y) &= E[XY] - E[X]E[Y] = E[X\frac{X - X_{\min}}{\overline{X}}] - E[X]E[\frac{X - X_{\min}}{\overline{X}}] = \\ &= \frac{1}{\overline{X}}\left(E[X^2] - X_{\min}E[X]\right) - \frac{1}{\overline{X}}\left(E[X]^2 - X_{\min}E[X]\right) = \\ &= \frac{1}{\overline{X}}\left(E[X^2] - E[X]^2\right) = \frac{1}{\overline{X}}\mathrm{cov}(X,X).\end{aligned} \tag{14}$$

We see that the transformation is scaled with a factor $\frac{1}{\overline{X}}$ which is the difference between largest and smallest element for each feature, thus the scaling is not constant. On the other hand we can interpret the covariance as the correlation scaled with the standard deviation for each feature, the correlation is unchanged by the transformation so this means that the new covariance matrix should behave qualitatively as

$$\mathrm{cov}(X,Y) \sim \frac{\sigma_X\sigma_Y}{\overline{X}}\mathrm{cov}(X,X). \tag{15}$$

While in most cases the scaling factor will be proportional to one (since a large standard deviation suggests a large $\overline{X}$) this is of course not true in the general case. E.g. a vector can be constant for all entries except for one very small and one very large outlier, resulting in a small standard deviation but a large $\overline{X}$.

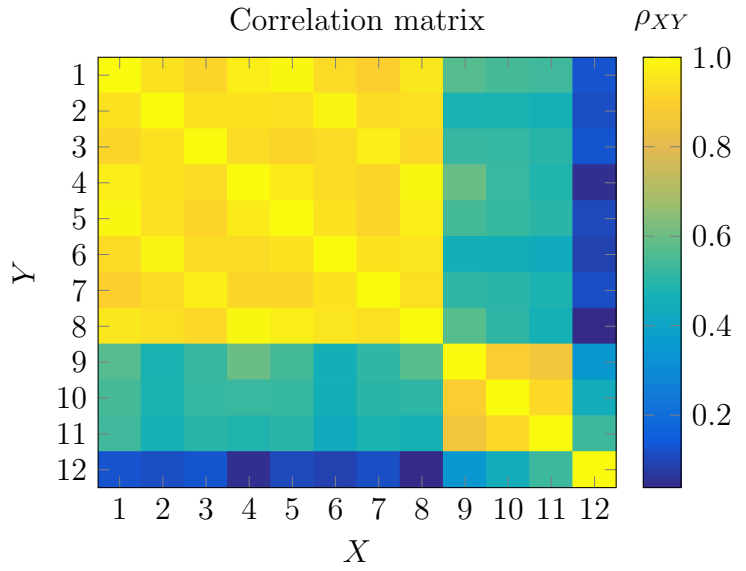


Figure 2: Colour map plot of the $\text{corr}(X, Y)$ i.e. the correlation between each feature is illustrated with a colour according to the colour bar to the right. N.b. that the diagonal is exactly equal to one suggesting that the transformation does not in fact affect the correlation.

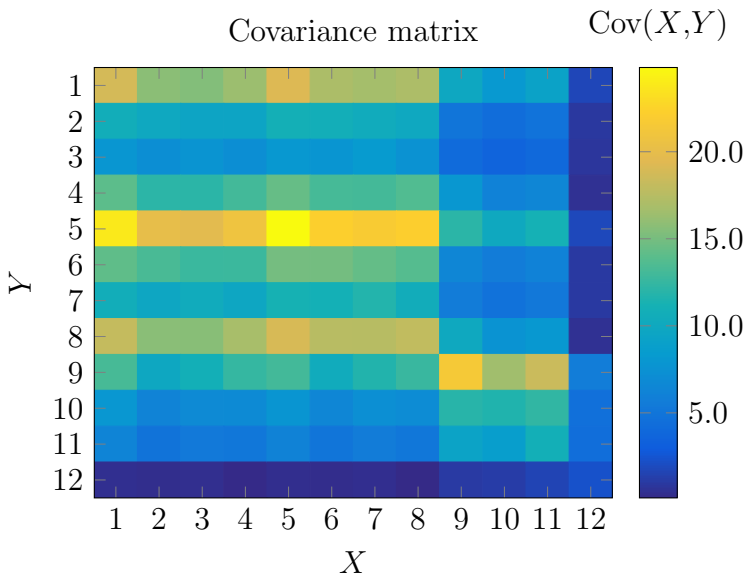


Figure 3: Colour map plot of the $\text{cov}(X, Y)$ i.e. the covariance between each feature is illustrated with a colour according to the colour bar to the right.

Lowest correlation between $Y_8$ and $Y_{12}$, $\min_{i,j\in\{1,\ldots,12\}}(\rho_{Y_iY_j}) = (\rho_{Y_8Y_{12}}) = 0.037987$

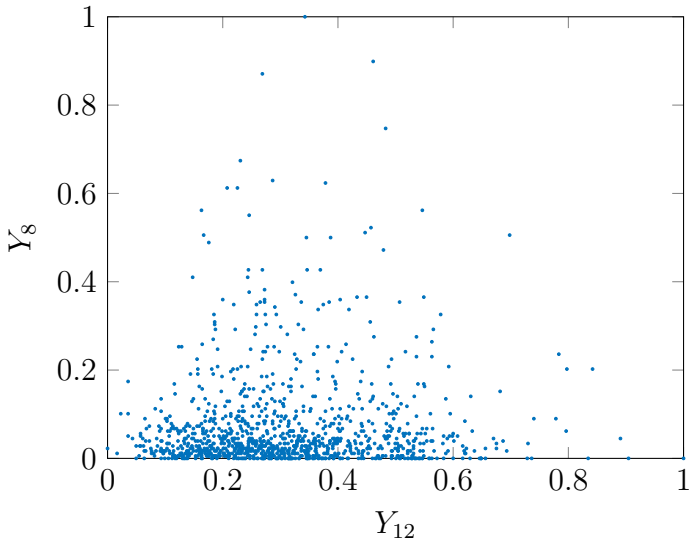


Figure 4: Scatter plot of the pair of features in Y with the lowest correlation, $Y_8$ and $Y_{12}$, with $\text{corr}(Y_8, Y_{12}) \approx 0.38$

# References

[1] Rogers, S. (2016). First Course in Machine Learning + Ebook. 1st ed. [S.l.]: CHAPMAN & HALL CRC.

[2] Devdatt Dubhashi *Lecture notes*

# A Matlab code

```
%% 2.1
% Parameters of the distr.
mu = [1;1];
Sigma = [0.1 -0.05; -0.05 0.2]
n = 1000; % Sample size
% Only need to calculate the inverse once.
S = inv(Sigma);

% Generate sample
X  = mvnrnd(mu,Sigma,n);

% Ezloop requires two inputs so we split here
x = X(:,1);
y= X(:,2);

% Preallocate
f3 = zeros(length(X),1);


% Obv. faster to use arrayfun but since the sample size is so small we
% allow ourselves the luxury of laziness and use a for loop.
for i = 1:length(X)
f3(i) = Gauss2D(x(i),y(i),3);
end

% Divide index according to limit f(x,3)>0
ind = find(f3>0);
indC = setdiff(1:length(X),ind);

figure(2)
clf
hold on
col = {'r','g','c'}
for r = 1:3
   %Generate contour plots
   h(r) = ezplot(@(x,y)Gauss2D(x,y,r),[-1 3]);
   h(r).LineColor = col{r}
end

% Plot points in- and outside level curve stated above.
plot(x(indC),y(indC),'.')
plot(x(ind),y(ind),'k.')
legend('f(x,1) = 0','f(x,2) = 0','f(x,3) = 0','f(x,1) < 0','f(x,1) > 0')
str = strcat('Number of points outside f(x,3) = ',num2str(length(ind)) );
title(str)

%% 2.2
% Data from webpage.
X = load('dataset0.txt');
% Transform data to values between 0 and 1.
Y = (X-min(X))./(max(X)-min(X));

% Actually simpler to use formal definition of cov to compute. Need to use
% bsxfun since mismatch of matrix dims. (Want row wise mult)
```

```
covXY = bsxfun(@minus,X,mean(X))'*bsxfun(@minus,Y,mean(Y))/(size(X,1)-1);
% Standard matlab function for corr.
corrXY = corr(X,Y);

% Plot
figure(3)
clf
imagesc(covXY)
colormap jet
title('Cov(X)')
axis equal
figure(4)
imagesc(corrXY)
colormap jet
title('Corr(X)')
axis equal
%%
% Generate data for tikz plotting in latex
clc
figMat = zeros(144,3);
for i= 1:length(covXY)
    for j=1:length(covXY)
        figMat(12*(i-1)+j,:) = [j i covXY(i,j)];
    end
    disp(figMat(12*(i-2)+j+1:12*(i-1)+j,:))
end
%%
corrYY = corr(Y,Y)
% Find the two features Y_i and Y_j with minimum correlation.
[rho indrows] = min(corrYY);
[rhoMin indcol] = min(rho)
indrow = indrows(indcol)
figure(3)
clf
plot(Y(:,indcol),Y(:,indrow),'.')
str = strcat('Lowest correlation between Y',num2str(indrow),' and Y',...
    num2str(indcol),', min(\rho) = ',num2str(rhoMin));
title(str)


function [ f ] = Gauss2D( x,y,r )
    mu = [1;1];
    % Sigma = [0.1 -0.05; -0.05 0.2]
    S = 1/(0.02-0.05^2)*[0.2 0.05; 0.05 0.1];
    x = [x y];
    f = (x'-mu)'*S*(x'-mu)/2 -r;
end
```