

# COMPSCI 527- Homework 1

Due on September 9, 2014

---

Questions may continue on the back. Type your text. You may write math formulas by hand. What we cannot read, we will not grade.

You may talk about this assignment with others, but do not write down anything while you talk. After that, do the assignment alone. What you hand in must be your work only.

Hand in your solution as a single, stapled paper document at the beginning of class on the due date.

The exercises in this homework assignment cover basic material on probability corresponding to chapter 2 of the textbook. This material has been briefly summarized in class, under the assumption that most of these concepts are familiar to you. If you have had probability in previous classes, you may be able to complete this assignment without reading the book. However, please refer to the textbook for notation, which may be different from what you have used in the past.

The difficulty of these problems varies, and I would expect you to take a few hours to complete the assignment. If it takes you an inordinate amount of time and effort to do the exercises (especially 1–5), or if a large number of the concepts involved are entirely foreign to you, you may want to reconsider taking this class.

Some questions require writing simple MATLAB code. Please intersperse concise comments with your code whenever useful, but do not go overboard with comments. Try to use matrix notation whenever possible, so as to avoid explicit `for` loops over images. These are very inefficient in MATLAB.

1. Compute the expected value of the fluctuation  $|X - m|$  from the mean  $m$  of a random variable  $X$  with uniform distribution between 0 and 1. Show your calculations. [Hint: The average fluctuation is not the standard deviation.]

2. Problem 2.4 from the book.

3. Problem 2.6 from the book. While this result is trivial to obtain, it gives a more intuitive meaning to the notion of statistical independence:  $x$  and  $y$  are independent if the conditional distribution of  $x$  given  $y$  is the same as the unconditional distribution of  $x$ . In other words, knowing  $y$  does not change the distribution of (does not change our information about)  $x$ . This of course goes both ways, as the definition of independence is symmetric in  $x$  and  $y$ .

4. Problem 2.8 from the book.

5. Problem 2.10 from the book.

6. In this set of exercises, you will implement a few MATLAB functions to work on joint probability distributions of two discrete random variables  $x$  and  $y$ . If  $x$  can assume  $m$  values  $\xi_1, \dots, \xi_m$  and  $y$  can assume  $n$  values  $\eta_1, \dots, \eta_n$ , we represent such a distribution with a  $m \times n$  matrix  $P$  of double numbers. Entry  $P(i, j)$  of this matrix is the probability that  $x$  equals  $\xi_i$  and  $y$  equals  $\eta_j$ . Of course, not every matrix  $P$  represents a valid probability distribution.

(a) Write a MATLAB function `isProbability` that takes a matrix  $P$  as input and returns `true` if  $P$  represents a valid probability distribution and `false` otherwise. [Hint: The entries of  $P$  are represented in finite precision, so equalities can only hold approximately.]

(b) Write the outputs from the following commands:

```
isProbability([0 1; 1 0])
isProbability([0 -0.2; 0.7 0.5])
isProbability([1 2 1; 3 0 1] / 8)
```

(c) Write a MATLAB function with header

```
function [Px, Py] = marginals(P)
```

that takes a matrix  $P$  that represents the joint probability distribution of  $x$  and  $y$  and computes the marginal probability distributions of  $x$  and  $y$ . If  $P$  has size  $m \times n$ , then  $P_x$  is a column vector with  $m$  entries and  $P_y$  is a row vector with  $n$  entries. For safety, your code should check that the input represents a valid probability distribution, and generate an error otherwise. [Hint: Do not confuse  $x$  and  $y$  with each other.]

(d) Write the outputs from the following commands:

```
P = [0 1 2 1; 0 9 0 3] / 16;
[Px, Py] = marginals(P)
```

(e) Write a MATLAB function with header

```
function [Pxgy, Pygx] = conditionals(P)
```

whose input argument  $P$  represents the joint probability distribution of  $x$  and  $y$ . The function computes two matrices<sup>1</sup>  $P(x|y)$  and  $P(y|x)$  each the same size as  $P$ . Column  $j$  of  $P(x|y)$  is the conditional probability  $\Pr(x | y = \eta_j)$  for all  $x$ . Row  $i$  of  $P(y|x)$  is the conditional probability  $\Pr(y | x = \xi_i)$  for all  $y$ . Again, check that the input represents a valid probability distribution, and do not confuse  $x$  with  $y$ .

There is a twist in this problem, in that the definition of conditional probability involves a division, and we need to avoid dividing by zero. For instance, if we compute

$$p(x|y) = \frac{p(x, y)}{p(y)}$$

for the matrix

$$p(x, y) = \begin{bmatrix} 0 & 0.2 & 0.3 \\ 0 & 0.4 & 0.1 \end{bmatrix}$$

we obtain the marginal distribution

$$p(y) = [0 \quad 0.6 \quad 0.4]$$

so that the two entries of the conditional probability  $p(x|y = \eta_1)$  are undefined (here,  $\eta_1$  is the first of two values that the random variable  $y$  can take).

In situations like these, we fill the corresponding column with a uniform probability distribution. There are  $m$  entries in a column if  $P$  is an  $m \times n$  matrix, so the uniform distribution has value  $1/m$  everywhere. The uniform distribution is said to be *uninformative*, in the sense that it carries the minimum possible amount of information about a random variable: we have no preference for one value of  $x$  over another, given that  $y = \eta_1$ . So the resulting matrix in this example is as follows:

$$p(x|y) = \begin{bmatrix} \mathbf{1/2} & 1/3 & 3/4 \\ \mathbf{1/2} & 2/3 & 1/4 \end{bmatrix}$$

(the uniform distribution is shown in bold).

Similar considerations hold for the rows of  $p(y|x)$ . In your code, it is easiest to write a subroutine that handles  $p(x|y)$  correctly, and then use matrix transposition appropriately to call the same subroutine to compute  $p(y|x)$ . Following this suggestion is optional, but saves you time and effort.

(f) Write the outputs from the following command

```
[Pxgy, Pygx] = conditionals(P)
```

where  $P$  is the same matrix you used to test marginals.

(g) One way to write Bayes's theorem is as follows:

$$\Pr(y = \eta_j | x = \xi_i) = \frac{\Pr(x = \xi_i | y = \eta_j) \Pr(y = \eta_j)}{\sum_j \Pr(x = \xi_i | y = \eta_j) \Pr(y = \eta_j)}.$$

In terms of the MATLAB matrices defined in earlier questions, this equality can be rewritten as follows:

$$\text{Pygx}(i, j) = \frac{\text{Pxgy}(i, j) \text{Py}(j)}{\text{Pxgy}(i, :) \text{Py}}$$

(if you remember what the entries of  $\text{Pygx}$  mean, you will realize that the order of  $i$  and  $j$  in  $\text{Pygx}(i, j)$  is not a typo). In the expression above,  $\text{Pxgy}(i, :)$  is MATLAB notation to denote the  $i$ -th row of  $\text{Pxgy}$ .

The matrix that contains all the entries in the left-hand side of this equation is obviously  $\text{Pygx}$ . Write a MATLAB function

```
function Pygx = bayes(Pxgy, Py)
```

that computes the matrix  $\text{Pxgy}$  in terms of the matrices  $\text{Pygx}$  and  $\text{Py}$ . As before, both  $\text{Pxgy}$  and  $\text{Pygx}$  are  $m \times n$  matrices and  $\text{Py}$  is a row vector with  $n$  entries. Check for input argument validity in your code (careful!), and prevent division by zero similarly to what you did for the function `conditionals`.

<sup>1</sup>You can read the MATLAB variable names  $\text{Pxgy}$  and  $\text{Pygx}$  corresponding to the mathematical symbols  $P(x|y)$  and  $P(y|x)$  as “probability of  $x$  given  $y$ ” and “probability of  $y$  given  $x$ .”

- (h) Evaluate the matrix `Pygx` with the function `bayes` you wrote as answer to the last question using the numerical values for `Pygx` and `Py` obtained in your answers to questions (d) and (f) above. Compare the resulting matrix with the matrix `Pxgy` you obtained in (f).
- (i) How can you use your function `bayes` to compute  $p(x|y)$  from  $p(y|x)$  and  $p(x)$ ? Just write the one line of code that does this, assuming that  $p(x|y)$ ,  $p(y|x)$ ,  $p(x)$  correspond to MATLAB variables `Pxgy`, `Pygx`, `Px`.

7. The image `yeast.png` available with this assignment shows several yeast cells viewed through a Differential Interference Contrast (DIC) microscope. Look at the image, either in MATLAB or in your favorite image viewer. Each cell is surrounded by a ring that is darker than both cell and background. In addition, a halo that is brighter than both cell and background is often visible on the lower left side of the cell. These are artifacts of the DIC imaging process, and we will call them the *dark ring* and the *bright halo* in this exercise.

A computer vision expert is asked to write software to segment images like `yeast.png`, that is, to create a binary mask image the same size as the input image. A pixel in the mask image is `true` (or 1) if the same pixel in the input image is on the body of a cell, and is `false` (or 0) otherwise. The expert's first thought is to segment the image with a threshold, under the assumption that cell bodies are either all brighter or all darker than the background. Pixels whose values in the input image are on one side of the threshold are set to `true` and the others to `false`.

To test this and other ideas, the expert labels the `yeast.png` image by hand. That is, he manually draws the desired output mask for this image using Adobe Photoshop. He then collects separate statistics of the pixel values in the cell bodies and in the background of the original image, using the mask as a guide. More specifically, he obtains the conditional probability  $p(x|c)$  that a pixel has value  $x$  given that it is in the cell, and the conditional probability  $p(x|b)$  that a pixel has value  $x$  given that it is in the background. In this and other distributions in this exercise, pixel values are in the set  $\{0, \dots, 255\}$ . The image has size  $1024 \times 1024$ , and 58,299 pixels were found to be on cell bodies.

- (a) A pixel in `yeast.png` is either on a cell or in the background. What are the probabilities  $p(c)$  and  $p(b)$  that a pixel is on a cell or on the background, respectively, as estimated from the image `yeast.png`? Use three significant (nonzero) decimal digits in your answer.
- (b) The conditional probabilities  $p(x|c)$  and  $p(x|b)$  found by the expert are contained in variables `pxgc` and `pxgb` in the file `hw1.mat` available with this assignment. You can load this file in MATLAB with the command

```
load hw1
```

and the variables will then be defined in your MATLAB workspace. Plot the two conditional probabilities on the same diagram using the MATLAB `stairs` function. Use different colors for the two plots, label the axes with values and meaning of the axes, and add a legend on the graph that tells which plot is which. Hand in your plot (not the code you write to generate it). [Hint: The MATLAB functions `xlabel`, `ylabel`, and `legend` and the command `hold on` are useful here. MATLAB has a very good help facility to find out how to use these and other utilities (look for a question mark in the menu bar to start help).]

- (c) Compute the conditional probabilities  $p(c|x)$  and  $p(b|x)$  that a pixel drawn at random from the image belongs to a cell or to the background, given that the value of the pixel is  $x$ . These two functions of  $x$  are called the *posterior probabilities* of cell/background given pixel value. Use the MATLAB `stairs` function to plot both  $p(c|x)$  and  $p(b|x)$  as functions of  $x$  on the same diagram and in different colors. Show the code you wrote to compute the posteriors, and show your figure. As before, add meaningful axis labels and legends. [Hint: You may want to use the `bayes` function you developed earlier, although this is not mandatory.]
- (d) If you did things right, the two plots in your answer to the last question should be mirror images of each other. Why?
- (e) The plots of  $p(c|x)$  and  $p(b|x)$  do not seem to reveal a natural place to put a threshold  $\xi$  such that all pixel values  $x$  that are on one side of  $\xi$  are cells and all others are background. This is because the cell bodies have bright pixels (the bright halos, plus some specs in the middle), dark pixels (the dark rings), and pixel values in-between. The pixel values in the background are in the middle of the range of brightnesses, so there exists no single threshold on pixel values that can distinguish cell bodies from background.

Instead of putting the threshold on the pixel value  $x$ , we can instead define a threshold  $\tau$  on  $p(c|x)$ , so that pixels whose value  $x$  satisfies  $p(c|x) > \tau$  are declared to be cell pixels. What is the most natural value of  $\tau$ , given the definition of  $p(c|x)$ , and why?

- (f) You have just come up with what is called the *Bayes classifier* for our segmentation problem. Try it on the image `yeast.png` and show the resulting mask image. Avoid complicated and slow code with explicit `for` loops. As an example of “thinking in MATLAB,” I’ll show you how to write this code. You may want to revisit some of your previous answers if you use explicit loops in them. In the following code snippet, I assume that your image is in variable `img`, your posterior distribution  $p(c|x)$  is in variable `pcgx`, and your threshold is in variable `tau`.

```
bayesCell = pcgx(img) > tau;  
imagesc(bayesCell)  
colormap gray  
axis image  
axis off
```

You can then use the MATLAB `print` command to save the result in your favorite file format.

- (g) Comment on your result, which are inevitably far from perfect. In particular, what does the classifier flag as “cells”? Is this satisfactory? Is it a good start? Are there ways to get much better results based on individual pixel values alone? Why or why not?
- (h) In what way is the Bayes classifier more expressive than the classifier based on a single threshold on pixel value  $x$ ? Another way to ask this question is as follows: Describe the types of subsets of  $X = \{0, \dots, 255\}$  that each classifier can yield in principle, as  $p(c|x)$  is suitably changed.