Tutorial 3: Naive Bayes and Gaussians

1. This question is similar to the example done in Learning and Data notes section 7.4, except that the data set is slightly different.

   Consider a set of documents each of which is related either to *Sports* (S) or to *Informatics* (I). Given a training set of 11 documents, we would like to estimate a Naive Bayes classifier, using the multinomial document model, to classify unlabelled documents as S or I.

   We define a vocabulary of eight words:

$$V = \begin{bmatrix} w_1 = \text{goal}, \\ w_2 = \text{tutor}, \\ w_3 = \text{variance}, \\ w_4 = \text{speed}, \\ w_5 = \text{drink}, \\ w_6 = \text{defence}, \\ w_7 = \text{performance}, \\ w_8 = \text{field} \end{bmatrix}$$

   Each document $\mathcal{D}_i$ is represented as an 8-dimensional vector $\mathbf{m}_i$; element m_{it} is the frequency of word w_t in document $\mathcal{D}_i$. The training data is presented below as a matrix for each class, in which each row represents an 8-dimensional document vector.

$$\mathbf{M}^{\text{Sport}} = \begin{pmatrix} 3 & 0 & 0 & 0 & 1 & 2 & 3 & 1 \\ 0 & 0 & 1 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 2 & 1 & 0 \\ 1 & 0 & 0 & 2 & 0 & 1 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 0 & 1 & 3 \\ 0 & 0 & 1 & 2 & 0 & 0 & 2 & 1 \end{pmatrix}, \quad \mathbf{M}^{\text{Inf}} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

   Estimate the parameters of a multinomial model for the two document classes, for two cases, case 1: we do not use any smoothing; case 2: we use add-one smoothing. Classify the following test vectors into Sports or Informatics for each case.

   (a) $\mathcal{D}_1 = w_5\ w_1\ w_6\ w_8\ w_1\ w_2\ w_6$

   (b) $\mathcal{D}_2 = w_3\ w_5\ w_2\ w_7$

2. This question considers writing program code for simulating random events such as coin tossing and dice rolling. To generate random values, we use a function, rand(), which returns a random real value (scalar) uniformly distributed on the open interval (0,1).

   Write the pseudo code of simulating each experiment shown below:

   (a) Toss a fair-coin ten times, and shows the outcome - the sequence of the faces observed.

   (b) Toss an unfair-coin ten times, where $P(H) = 0.6$ and $P(T) = 0.4$, which represent the probabilities of getting heads and getting tails, respectively.

(c) Roll an uneven die of six faces ten times, where $P(1)=0.05, P(2)=0.1, P(3)=0.14, P(4)=0.19, P(5)=0.24, P(6)=0.28$.

3. (a) Sketch, on the same graph, Gaussian pdfs with the following values of mean (μ) and variance (σ^2):
   i. $\mu = 0;\ \sigma^2 = 1$
   ii. $\mu = 2;\ \sigma^2 = 1$
   iii. $\mu = -2;\ \sigma^2 = 1$
   iv. $\mu = 0;\ \sigma^2 = 10$
   v. $\mu = 0;\ \sigma^2 = 0.1$

   (b) Describe how the width and height of the curve depends on the variance.

   (c) Write Matlab/Python code to draw the Gaussian pdfs.

4. Consider a two-dimensional Gaussian distribution $N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, whose parameters are estimated from the following training samples:

$$(1,1)^T; \quad (1,2)^T; \quad (3,2)^T; \quad (3,3)^T$$

   (a) Estimate the mean vector $\hat{\boldsymbol{\mu}}$ and the covariance matrix $\hat{\boldsymbol{\Sigma}}$ based on the maximum likelihood criterion.

   (b) Find the correlation matrix using the covariance matrix obtained in (a).

   (c) Sketch the contours of the Gaussian distribution in a 2D plain.

   (d) Find the log likelihood of a new data point $\mathbf{z} = (2,1)^T$.

   (e) Assume you had only two training samples, $(1,1)^T$ and $(3,2)^T$, discuss possible problems you might have in (a), (b), (c), and (d) above.

5. For a dataset of n points $\{x_i\}_1^n$, the maximum likelihood estimators of mean and variance are given by

$$m = \frac{1}{n}\sum_{i=1}^{n} x_i$$
$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - m)^2 .$$

It often happens that new points are added to our dataset, and in this case it is very useful to update the mean and variance rather than recalculate it for the entire sample.

Verify that the following algorithm results in `m` as the mean and Sn as n times the variance:

```
Initialise:  Sn <- 0; m <-0
For i = 1 to n do:
   r(i) <- (x(i) - m)
   Sn <- Sn + (1 - 1/i) r(i)^2
   m <- m + r(i)/i
endfor
```