STAT0030 Assessment 3 — Instructions

For this assessment you should submit online – on the course Moodle page using the link "ICA3: Click here to submit your assignment". Make sure none of the files contains your surname, as the marking must be anonymous. You must submit two files:

- An electronic copy of your `StudentNumber.rmd` file, containing your R markdown code. For example, if your student number is 18239004, your R markdown script should be saved in the file `18239004.rmd`.
- A single PDF file named `StudentNumber.pdf` containing the knitted output of the Rmarkdown file. This should correspond **exactly** to what is produced when knitting the submitted `.rmd` file.

Any output within your pdf should be clearly presented and structured according to the question parts.

STAT0030 Assessment 3 – Marking guidelines

The assessment is marked out of 100, with Questions 1, 2 and 3 having a total of 30, 20 and 50 marks, respectively. The marks for each question are subdivided into the following components.

1. For Question 1, items (a), (b) and (c) have a breakdown of 15, 10 and 5 marks, respectively.
2. For Question 2, items (a), (b) and (c) have a breakdown of 2, 8 and 10 marks, respectively.
3. For Question 3, items (a), (b), (c), and (d) have a breakdown of 25, 5, 5, 15 marks, respectively.

For each coding part, graphical presentation (appropriate choice of graphs and formatting), quality of printed output (appropriate messages printed) and quality of the code (your code should be clean, readable – with sufficient commenting for the user – and efficient) are factors to be considered.

STAT0030 Assessment 3 — Questions

Background. Clustering and dimensionality reduction are often seen as complementary tasks in unsupervised learning, but they can also be combined. One approach is the *mixture of factor analysers*[1]. The idea is to start with the usual mixture of Gaussians (as seen in Section 2.3 of Lab 8), but where each Gaussian is a probabilistic variant of PCA. What does it mean? We will start first by describing the basic factor model.

Factor models. Say we want to model the distribution of some iid data points $\mathbf{X}_1, \ldots, \mathbf{X}_N$, where each data point $\mathbf{X}_i$ is a vector with entries $X_{i1}, X_{i2}, \ldots X_{iJ}$. One idea is to assume that there are also unobserved variables $\mathbf{Y}_i$, a vector with entries $Y_{i1}, Y_{i2}, \ldots, Y_{iM}$, such that

$$X_{ij} = \mu_j + \sum_{m=1}^{M} b_{jm} Y_{im} + \epsilon_{ij}.$$

If each ϵ_{ij} is independent and a zero-mean Gaussian with variance σ_j^2, and all Y_{im} are also independent and standard Gaussian random variables, we can show that

$$\begin{aligned} \mathbb{E}[\mathbf{X}_i] &= \mu, \\ \text{Cov}[\mathbf{X}_i] &= BB^\mathsf{T} + D, \end{aligned}$$

where μ is the J-dimensional vector with entries $\mu_1, \ldots, \mu_J$; B is a $J \times M$ matrix with b_{jm} being the (j, m) entry of B; and D is a $J \times J$ diagonal matrix, where entry (j, j) is σ_j^2.

The likelihood function is therefore a function of μ, B and the diagonal of D. The dataset is $\{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$, since no $\mathbf{Y}_i$ is ever observed. The relationship between the Gaussian factor model and PCA is sometimes explained in terms of the conditional expectation $\mathbb{E}[\mathbf{Y}_i \mid \mathbf{x}_i]$. Under some conditions on D, this conditional expectation is the same as the PCA projection[2]. The use of factor models in statistical inference is sometimes called *factor analysis*, with *factor analyser* being sometimes used a synonym for factor model.

The mixture of factor analysers. Like in mixture of Gaussians, we have that each observation $\mathbf{X}_i$ has an "allocation indicator" Z_i that here changes the relationship between $\mathbf{X}_i$ and $\mathbf{Y}_i$. Like a cluster label, Z_i is assumed to take values in $\{1, 2, \ldots, K\}$ for some chosen K. We can describe how one would simulate a data point $\mathbf{X}_i$ as follows:

1. Sample Z_i from a categorical distribution in $\{1, 2, \ldots, K\}$ with parameter π, a K-dimensional vector where $P(Z_i = 1) = \pi_1, \ldots, P(Z_i = K) = \pi_K$;
2. Sample $\mathbf{Y}_i$ as M independent standard Gaussian random variables;
3. Sample $\mathbf{X}_i$ according to the following formula, independently for each $X_{i1}, \ldots, X_{iJ}$:

$$X_{ij} = \mu_j^{(Z_i)} + \sum_{m=1}^{M} b_{jm}^{(Z_i)} Y_{im} + \epsilon_{ij}, \text{, where } \epsilon_{ij} \sim \mathcal{N}(0, \sigma_j^2).$$

[1] Z. Ghaharamani and G. Hinton, 1997, `https://www.cs.toronto.edu/~hinton/absps/tr-96-1.pdf`

[2] M. Tipping and C. Bishop (1999). "Probabilistic principal component analysis". J. R. Statist. Soc. B (61), 611–622.

The likelihood function is given by π; the set of mean parameters $\mu^{(1)}, \ldots, \mu^{(K)}$, each a vector of dimension J; the set of matrices $B^{(1)}, \ldots, B^{(K)}$, each a $J \times M$ matrix with $b_{jm}^{(k)}$ being the (j, m) entry of $B^{(k)}$; and D, a diagonal matrix of size $J \times J$, where entry (j, j) is σ_j^2.

As in the mixture of Gaussians, the likelihood function at each data point $\mathbf{x}_i$ is given by:

$$f(\mathbf{x}_i \mid \theta) = \sum_{k=1}^{K} f(\mathbf{x}_i \mid Z_i = k, \theta) P(Z_i = k \mid \theta) = \sum_{k=1}^{K} \pi_k g_k(\mathbf{x}_i \mid B^{(k)}, \mu^{(k)}, D).$$

Here, θ is the set of all parameters and g_k is a multivariate Gaussian density function with mean $\mu^{(k)}$ and covariance $B^{(k)} B^{(k)^\mathsf{T}} + D$.

Questions. The following questions will use the mixture of factor analysers package `EMMIXmfa`, which can be installed in the usual way by `install.packages("EMMIXmfa")`. All exercises will be based on the pre-loaded dataset `iris`. You can check its documentation using `?iris`. In all that follows, we will use function `mfa` from `EMMIXmfa`. Read the documentation using `?mfa`, where an example with `iris` is already shown. *It is your job to understand the documentation, including some differences in notation.* Setting `nkmeans = 3` and `nrandom = 0` in all calls to `mfa` will speed things up and be enough for `iris`, and for this exam you MUST use this setting to make sure the submitted code is reproducible.

1. This question concerns fitting and evaluating a `mfa` model.

   (a) Fit a mixture of factor analysers with $M = 2$ and $K = 3$ for the first four columns of `iris`. Don't scale the data. Make sure that the model uses a different B matrix for each mixture component (which is *not* the default of `mfa`). From the fitted model, extract all the corresponding parameters θ to simulate data based on the model. You need to program the simulation explicitly, i.e., your code should be based on the sampling procedure of the previous page. You are allowed to use functions `sample` and `rnorm`. You must provide your code in terms of a function `simulate_mfa` that takes as arguments `N`, the number of samples; and `mfa_model`, the output of a call to `mfa`. The function must output a $N \times J$ matrix of your samples. Apply this function to the fitted model to simulate a sample of the same sample size as `iris`.

   (b) Visually compare the result of this simulation against the real data to provide an informal goodness-of-fit assessment. Plots of interest include quantile-quantile plots and scatterplots. Use your judgement and write one short paragraph (recommended maximum of 200 words) with your conclusions.

   (c) Without providing any code, explain briefly how you would do model selection to decide on a choice of K and M (recommended maximum of 200 words).

2. Let us now use the model for clustering (under the hood, the logic is the same as in a mixture of Gaussians: for each $\mathbf{x}_i$: the cluster assignment is decided by the value z_i that maximises $P(Z_i = z_i \mid \mathbf{x}_i)$). As in the previous question, keep $K = 3$ and $M = 2$.

(a) Use function `predict` to generate cluster assignments from the output of `mfa` as applied to the first four columns of `iris`. Run `kmeans` on the same, unscaled, columns of `iris`, again using 3 clusters. We want to evaluate the agreement between the two clusterings. However, given that this is unsupervised learning, the cluster *labels* for both `kmeans` and `mfa` are arbitrary. Function `ari` can be used to compare classifications with arbitrary labels, so use it to report the agreement between the two clusterings, and each clustering against the labels reported in column `Species`.

(b) Discuss the results obtained, and how you explain the differences. Describe one way by which you could improve the agreement between the `kmeans` selection and the true labels, and why it would work (recommended maximum of 200 words).

(c) Generate a new dataset `iris_noisy`, made of the first four columns of `iris` and ten columns of independent standard Gaussians. Use the `randomForest` function from the respective package seen in Lab 7 to learn and evaluate the ability of this model to predict `iris$Species` from `iris_noisy`. To simplify, there is no need to fiddle with hyperparameters or test sets: just look at the out-of-bag error evaluation. Report on the difference between the random forest evaluation and what the `ari` function reports for `mfa` also applied to this modified dataset. Provide a brief explanation for the differences obtained (recommended maximum of 200 words).

3. One interesting property of having a model (as in the mixture of factor analysers) as opposed to just the solution of a projection problem (as in PCA) is that the model has a natural way of dealing with missing data. In particular, clustering is still applicable even if some variables in $\mathbf{x}_i$ are not recorded. Furthermore, we can use the model to estimate those missing records, a procedure sometimes known as *data imputation*.

(a) Using mathematical results provided at the Technical Hints section at the end of this document, write a function `mfa_impute` that takes two arguments: `dat`, a data frame which may have any possible combination of `NA` entries; and `mfa_model`, the output of a call to `mfa`.

Let K be the number of mixture components in `mfa_model` and let N be the number of rows in `dat`. Function `mfa_impute` computes, for every row $i = 1, 2, \ldots, N$ in `dat` and every mixture component value $z_i = 1, 2, \ldots, K$, the conditional distribution $P(Z_i = z_i \mid \mathbf{x}_{io_i})$. Vector $\mathbf{x}_{io_i}$ is the subvector of $\mathbf{x}_i$ without `NA` entries, which may vary with i. This function must also compute $\mathbb{E}[\mathbf{X}_i \mid \mathbf{x}_{io_i}, z_i]$ for every row i and value of z_i. As an output, the function must return a list with two entries.

The first element of the output must be called `P_Z`, a matrix of size $N \times K$ such that `P_Z[i, k]` corresponds to the computed value of $P(Z_i = k \mid \mathbf{x}_{io_i})$.

The second element of the output must be called `E_X`, and consist of a three-dimensional array of size $N \times J \times K$, where J is the number of columns in `dat`. Entry `E_X[i, j, k]` corresponds to $\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}, Z_i = k]$.

(b) Randomly partition `iris` into two datasets, `iris1` with 100 data points and `iris2` with the remaining 50 data points. Fit `mfa` to the data corresponding to the first four

columns of `iris1` using $K = 3$ and $M = 2$, storing the output in a variable called `mfa_model`. Modify `iris2` so that, for each row of `iris2`, you choose uniformly at random exactly one of its four columns to be set to `NA`, calling the resulting dataset `iris3`. Call `mfa_impute` with (the first four columns of) `iris3` and `mfa_model` as arguments, storing the result in a variable called `iris3_imputed`.

(c) For each row i in `iris3`, compute $\hat{Z}_i$, the position in `iris3_imputed$P_Z[i, ]` that has the highest value (so that each $\hat{Z}_i$ is in $\{1, 2, 3\}$). Store the result in a vector `Z_hat`. Using `predict`, generate the vector `Z_hat2` of cluster allocations for `mfa_model` and `iris2`. Using the `ari` function, print a comparison of how well each of `Z_hat` and `Z_hat2` agrees with the classification provided by `iris2$Species`. Provide a short comment about your findings (recommended max. of 100 words).

(d) For each entry X_{ij} of `iris3` that is missing, compute $\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}]$. For each column j, print the mean squared error between your estimate of the missing entries of that column and the ground truth in `iris2`. When calculating this error, ignore any row for which X_{ij} is observed. Compare how well you do against estimating each missing X_{ij} by just using the empirical average of the observed entries in `iris3[, j]`. Provide a short comment about your findings (recommended maximum of 100 words).

HOW TO PREPARE YOUR SUBMISSION. You must provide a single `.rmd` file including all of your code and answers to explanatory questions. **The file MUST contain exactly one chunk of code for each of the three questions, which implements and runs all the required steps in that question in a self-contained way. Each chunk MUST start with the line** `set.seed(12345)` **to make it reproducible.** The clarity of the printed output will be taken into consideration (e.g., in Question 3(d), you may want to print messages specifying which numbers correspond to which mean squared errors). It is up to your judgement to decide how the printed output should be organised. Answers to explanatory points within each question should appear in a single block of text, after the code and output for that respective question. Explanatory points should be presented at a level that can be understood easily by somebody with a MSc in Statistics.

Code comments should clarify which parts of each question are being solved. When submitting, knit the corresponding `.rmd` into a `.pdf`, including all code and output. Both the `.rmd` and the `.pdf` will need to be submitted.

For instance, for Question 1, prepare a single chunk of code which i) fits `iris` using `mfa`; ii) defines function `simulate_mfa`; iii) generates a simulated dataset; iv) provides the visual comparison between simulated and real data by your own judgement of what should be plotted. The code will be followed by a block of text with the explanatory aspects of Questions 1(b) and 1(c). For Question 1, the `.pdf` file will show all of that question's code, followed by its output, followed by its textual explanations. This will then be followed by the respective steps for Questions 2 and 3.

STAT0030 Assessment 3 — General hints

1. In general, there is not a single "right" answer to each question. To obtain a good mark you should approach the questions sensibly and justify what you're doing. Credit will be given for code that is clear and readable, while code that is inadequately commented will be penalised. You might like to use scripts `cosapprox.r` (Lab 1) and `tablet.r` (Lab 3) as models.

2. This assessment is designed to test your ability to understand data analysis algorithms in an applied context, their limitations and ways to combine then. This will be assessed not only on your computing skills, but also on your ability to carry out a critical assessment of what these models and algorithms can accomplish, using a simple dataset as an example. To earn high marks for this question, you need to take a structured and critical approach to the analysis and to demonstrate appropriate judgement in your choice of material to present.

3. Marks will be deducted if your `.pdf` file does not correspond *exactly* to the results we obtain when we knit the `.rmd`. You should *not* call `install.packages` anywhere in your file, but do all necessary calls to `library`. Marks will be deducted otherwise.

4. More credit will usually be given for code that is more generally applicable, rather than tailored to a particular situation or set of data. For example, if you were asked to print out the mean age of a group of people, you could do either of the following:

   - Calculate the mean before you write your final script, and then insert a line

     ```
     cat("Mean age is 25.3\n")
     ```

     (or whatever the mean happens to be) into your script.
   - In your script, create an object (say `xbar`) that holds the mean age, and then insert the line

     ```
     cat(paste("Mean age is",xbar,"\n"))
     ```

     into your script.

   The second approach is clearly more general and will earn more credit, since it will work for other similar data also.

5. All graphs should be clearly and appropriately labelled (giving units of quantitative variables), titled and formatted. By 'appropriately formatted' we mean, for example, that axis scales should be well chosen.

6. Your program should be **well commented**. If you have defined functions, these should consist of a header section summarising the logical structure, followed by the main body of the script. The main body should itself contain comments.

7. Refer to the feedback you received/will receive on in-course assessments 1 and 2.

STAT0030 Assessment 3 — Technical hints

1. For Question 3, you need the following results[3]. Let $\mathbf{X}_i$ be a random $J \times 1$ column vector following a multivariate Gaussian distribution for a given $Z_i = z_i$,

$$\mathbf{X}_i \mid z_i \sim \mathcal{N}(\mu^{z(i)}, \Sigma^{z(i)}),$$

where mean vector $\mu^{z(i)}$ is $J \times 1$ and covariance matrix $\Sigma^{z(i)}$ is $J \times J$. Then for any j in $1, 2, \ldots, J$ and some $o_i \subset \{1, 2, \ldots, J\}$, we have

$$\boxed{\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}, z_i] = \mu_j^{z(i)} + \Sigma_{jo_i}^{z(i)} [\Sigma_{o_i o_i}^{z(i)}]^{-1} (\mathbf{x}_{io_i} - \mu_{o_i}^{z(i)}),}$$

where $\Sigma_{o_i o_i}^{z(i)}$ and $\Sigma_{jo_i}^{z(i)}$ are the corresponding submatrices of $\Sigma^{z(i)}$, with an analogous idea for $\mu^{z(i)}$. Notice that $\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}, z_i] = x_{ij}$ if $j \in o_i$.

The other expectation $\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}]$ can be obtained from π and $\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}, z_i]$ using the law of iterated expectations,

$$\boxed{\mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}] = \sum_{k=1}^{K} P(Z_i = k \mid \mathbf{x}_{io_i}) \mathbb{E}[X_{ij} \mid \mathbf{x}_{io_i}, Z_i = k].}$$

Finally, to get $P(Z_i = z_i \mid \mathbf{x}_{io_i})$, we can just apply Bayes' rule:

$$\boxed{\begin{aligned} P(Z_i = z_i \mid \mathbf{x}_{io_i}) &\propto P(Z_i = z_i) f(\mathbf{x}_{io_i} \mid z_i) \\ &= \frac{\pi_{z_i} g(\mathbf{x}_{io_i} \mid \mu_{o_i}^{(z_i)}, \Sigma_{o_i o_i}^{(z_i)})}{\sum_{k=1}^{K} \pi_k g(\mathbf{x}_{io_i} \mid \mu_{o_i}^{(k)}, \Sigma_{o_i o_i}^{(k)})}, \end{aligned}}$$

where $g(\mathbf{x} \mid \mu, \Sigma)$ is the multivariate normal pdf with mean μ and covariance Σ evaluated at $\mathbf{x}$.

2. You are not expected to write very complicated algorithms for this assessment. Instead, we want you to be able to demonstrate your understanding of the logic and limitations of different methods and the contrast between supervised and unsupervised learning.

3. You are allowed to use R libraries for this assessment. However, you need to make sure that you fully understand what each function is doing and briefly explain it either in your writeup or in your code. Use of *any* libraries beyond what we have used in STAT0030 (or ones mentioned in these instructions) will result in a lower mark.

[3] The Wikipedia page on multivariate normals is particularly good with explanations, see `https://en.wikipedia.org/wiki/Multivariate_normal_distribution`.