Machine Learning (HWS24)
Assignment 4: Latent Variable Models

The archive provided to you contains this assignment description, a datasets in binary format, as well as Python code fragments for you to complete. Comments and documentation in the code provide further information. Please note the following:

- It **suffices to fill out the "holes" that are marked in the code fragments** provided to you, but feel free to modify the code to your liking. The aim of the assignments is to familiarize you with NumPy, so use NumPy over pure Python and try to keep your implementation efficient.
- Please **adhere to the following guidelines** in all the assignments. If you do not follow those guidelines, we may grade your solution as a FAIL. Provide a single ZIP archive with name `ml24-a0<assignment number>-<your ILIAS login>.zip`. The archive needs to contain:
  - A **single PDF report** that contains answers to the tasks specified in the assignment, including helpful figures and a high-level description of your approach. **Do not simply convert your Jupyter notebook to a PDF!** Write a separate document, stay focused and brief. You report **must not exceed 8 pages, excluding references and appendix**. You must use the seminar template from this link for your report (including the final signature page).
  - All **the code that you created** and used in its original format.
  - A **PDF document that renders your Jupyter notebook with all figures.** (If you don't use Jupyter, then you obviously do not need to provide this.)
- **A high quality report is required to achieve an EXCELLENT grade.** Such a report is self-explanatory (i.e. do not refer to your code except for implementation-only tasks), follows good scientific practice (e.g. when using images, tables or citations), does not include hand-written notes, and does not exceed 8 pages. In addition, label all figures (and refer to figure labels in your write-up), include references if you used additional sources or material, and use the tasks numbers of the assignments as your section and subsection numbers.
- You **may work on this assignment in pairs**—i.e, with one (and only one) additional student—and then hand in a pair submission. To do so:
  - The PDF report and notebook must clearly report then **name and ILIAS login** of **both students** right at the beginning.
  - **Both students must submit** the same assignment on ILIAS separately.
  - You may change whether or not you submit in a pair and with whom from assignment to assignment.

  To be clear, if only one student of the pair submits the assignment, then only this student will receive the grade.
- Hand-in your solution via ILIAS until the date specified there. **This is a hard deadline**.

Report. The following list of tasks are pure implementation tasks:

Task 1c, Task 2c

You do *not* need to discuss these tasks in your report.

1 Probabilistic PCA

a) Study the PPCA generator provided to you (`ppca_gen`). Generate and plot the `toy_ppca` dataset. What is shown in the plot? Vary the amount of noise (σ^2), replot, and describe how the data distribution changes (visually) depending on σ^2.

b) Implement MLE for PPCA by completing `ppca_mle`. Only use standard matrix/vector operations and the `svd` function. If we fit the PPCA model with $L = 2$ on `toy_ppca`, we obtain $\hat{\sigma}^2_{\text{MLE}} = 0$. Why is this?

c) Implement the computation of the conditional negative log-likelihood of a given dataset for a given PPCA model. To do so, complete `ppca_nll`.

d) Load the `secret_ppca` dataset, which was produced by sampling a PPCA model. Discover it's secret: How many latent variables have been used (L)? Do this by

   (i) Studying the scree plot.

   (ii) Using validation data.

   Do the results agree?

2 Gaussian Mixture Models

a) Study the GMM generator provided to you (`gmm_gen`). Generate and plot the `toy_gmm` dataset. Describe what is shown in the plot.

b) Compute a K-Means clustering of `toy_gmm` using $K = 5$ (code provided). Plot the resulting clustering and discuss. Did you expect this result?

c) Implement the E step and the M step of fitting a GMM with MLE by completing `gmm_e` and `gmm_m`. Only use standard matrix/vector operations and (optionally) the functions mentioned in the hints below.

   Hint (E step). Start by computing and verifying $\boldsymbol{F}$ ($[\boldsymbol{F}]_{ik} = f_k(\boldsymbol{x}_i)$ in lecture notation). You can compute the density of each data point (row) in a dataset ($\boldsymbol{X}$) under multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ using:

   ```
   dist = scipy.stats.multivariate_normal(mean=mu, cov=Sigma)
   densities = dist.pdf(X)
   ```

   Hint (M step). Start by computing and verifying $\boldsymbol{\pi}$, then the $\boldsymbol{\mu}_k$'s, then the $\boldsymbol{\Sigma}_k$'s. Perhaps helpful: NumPy's covariance function `np.cov` supports weighted data points (set `aweights` argument accordingly; additionally set `dof=0`).

d) Fit the GMM model with $K = 5$ using `gmm_fit` (provided). Assign each data point to its most likely component to obtain a clustering and plot the dataset as in subtask b). Compare the resulting clustering with the result of K-Means clustering of subtask b).

e) Repeat subtask d) with $K = 4$ and with $K = 6$. In both cases, repeat multiple times. Discuss your findings.

f) Optional: Load the `secret_gmm` dataset, which was produced by sampling a GMM model. Discover it's secret: How many components (L) have been used? Briefly justify your answer.