Machine Learning & Data Science II — Winter 2025/2026

5. Exercise sheet — 05. November 2025

Reminders to finish last exercises and to our way of working: -1, 0
Prepare before class such that you can present your result: 1, 2, 5
Worked on in class, but you can prepare them at home: 3, 4, 6

Carefully read the instructions below!

(-1) Finalize last exercise sheet

Finalize all exercises of the last exercise sheet and upload the results to your defined GitHub repository.

(0) Setup a git project on `GitHub`

In order to get a bit of `git` training done we work for all the exercises on GitHub.

(a) Use the project created for the previous exercises or create a new **private** project in GitHub

(b) Give the instructor (kandolfp) access to the project (see GitHub documentation for help)

(c) Create a `pdm` project in your repository and commit the necessary files.

(d) Create an appropriate structure in your repository for the rest of the exercise sheet (maybe have a look at the exercises to have a better idea first), not everything should be in the main folder.

(e) Try to structure your work on the exercises with git, i.e.

- Don't commit things that do not belong together in one single commit. Each exercise can be considered as a separate thing. Subparts of an exercise might be independent as well.
- Use meaningful commit messages `https://www.conventionalcommits.org/en/v1.0.0/`
- Make sure that you do not commit something that does not work - produces an error. If you have difficulties with an exercise you can also commit your best effort in this case.

(f) Add a README.md that explains what you are doing, how to run the exercises and anything else that is necessary (quick guide to `pdm`), maybe note your name somewhere.

(g) Optional: Work with issues, you can reference the issue in the commit message, GitHub documentation

(1) PCA for the penguins

The package `seaborn` provides a great way to visualize datasets with the function `pairplot`. Use it to visualize the feature of the palmerpenguins dataset.

Now apply principal component analysis PCA to it. Use `pairplot` again and discuss the differences.

(2) PCA for image denoising I

For this exercise we work with the MNIST dataset, see lecture for how to get it. You might want to limit the amount of images to better fit your PC.

As our aim is to remove noise from images, we first need to add it to our dataset. Follow the following steps, make sure you keep a copy of your ground truth.

- Initialize the random number generators such that you can reproduce your results.

- Scale the images to $[0, 1]$.
- Add some noise to it via `np.random.normal()`
- Visualize the original image and the noisy image next to each other.

With the noisy images we can now work to reduce the noise via PCA.

- Train a PCA with your noisy dataset
- Apply the PCA to your noisy dataset to get a new dataset, we call it y
- Apply the inverse PCA transform to y via `PCA.inverse_transform`
- Visualize the original images, the noisy images, and the denoised images constructed above.

(3) PCA FOR IMAGE DENOISING II

Instead of using a standard PCA use a `KernelPCA` with the following parameters as a start to denoise the digits.

```
from sklearn.decomposition import KernelPCA

kernel_pca = KernelPCA(
  n_components=400,
  kernel="rbf",
  gamma=1e-3,
  fit_inverse_transform=True,
  alpha=5e-3,
  random_state=42,
)
```

Compare try different values for γ, α, `n_components`, and discuss the results in comparison to the *normal* PCA from the previous exercise.

(4) DECISION TREES - SENSITIVITY TO ROTATION AND INITIAL STATE

For the following simple test set try to apply a `DecisionTreeClassifier` to the sets (X, y) and (X_{rot}, y) with different `random_state` values and check the results and quality of the trees.

```
X = np.random.rand(100, 2) - 0.5
y = (X[:, 0] > 0).astype(np.int32)

angle = np.pi / 4
rotation_matrix = np.array([[np.cos(angle), -np.sin(angle)],
                             [np.sin(angle), np.cos(angle)]])
X_rot = X @ rotation_matrix
```

Create a pipeline via `make_pipeline` that contains a standard scaler and the PCA to transform the dataset. Now apply again a decision tree to the newly transformed dataset, visualize and discuss the results.

(5) K-MEAN CENTERS

We stick to the MNIST dataset. Apply the `KMean` algorithm to it (make sure your results are reproducible) with 10 clusters.

Visualize the cluster centers, you can get them with `kmeans.cluster_centers_[i]`, they need to be reshaped with `reshape(28, 28)` and discuss the results. What is the difference to the *representatives* from the lecture.

(6) k-mean, DBSCAN and PCA

For the palmerpenguins dataset, try to get the three different classes via the k-mean and DBSCAN method. Now apply a standard scalar and PCA to the dataset and again use k-mean and DBSCAN to recover the classes.

Try to solve these problems on your own. If you get stuck, maybe some of your fellow students can help you. Why not write down your question in the course forum in SAKAI. Please try not to write back entire solutions or exchange solutions in the forum, this is not the idea and will backfire in the long run (also we might decide to delete such posts).