Assignment 3: Clustering models

In this assignment, you will implement the K-means method and apply it to image compression. Then you will use the Gaussian Mixture Models from the scikit-learn library and select the number of clusters K using the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC).

Work in groups of one, two or three and solve the tasks described below. Write a short report containing your answers, including the plots and create a zip file containing the report and your Python code.

Alternatively, write a Jupyter notebook including your code, plots, and comments. In this case, when you are finished editing, re-run all the cells to make sure they work and then convert your notebook into a pdf (using the print function). Submit both the .ipynb file and the .pdf file.

Submit your solution through the Canvas website.

In this assignment, you are **not allowed to use machine learning toolkits such as scikit-learn for the K-means implementation, and AIC, BIC computation.**

Deadline: December 19th

Part 1 Implement the K-means algorithm (5 pts)

Given an input data set with N data points, implement a K-means parameter estimation function that returns K centroids and N labels

```
def kmeans(x, K, n_init):
    # x: input data
    # K: number of centroids
    # n_init: the number of initial guesses for the centroids
    return centroids, labels
```

Run this algorithm with n_init initial guesses. Do you get the same result every time? How do you determine what the final result should be?

How do you choose what K value to use?

During the development, you can use the sklearn function datasets.make_blobs for generating clusters to test your algorithm

- centroids: the sample mean of each cluster
- n_init: the K-means algorithm is sensitive to the initialization of the centroids; n_init is the number of initializations you run; the result is chosen as the centroids corresponding to the lowest SSE (or other criteria)
- labels: the cluster index of each data point

Part 2 Apply the K-means algorithm to compress an image (3 pts)

Download an image and load it in Python. Now apply your algorithm to compress the image. Try different K's and discuss the results. Save the image as a .png file to the file system. Is it smaller after compression (compared to the original image in the same data format)? Why?

Part 3 Use AIC and BIC to choose K for Gaussian Mixture Models (2 pts)

Use the sklearn built-in function for Gaussian Mixture Models. Implement two functions to compute AIC and BIC, respectively. Choose the best number of clusters K on the breast_cancer data set using AIC and BIC and discuss the results.

```
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer().data
```