Labs
Machine Learning Course
Fall 2021

EPFL
School of Computer and Communication Sciences
Nicolas Flammarion & Martin Jaggi
www.epfl.ch/labs/mlo/machine-learning-cs-433

Problem Set 11, Dec 2, 2021 (K-Means Clustering)

1 Theory Questions

1.1 Vector Calculus

Let $f : \mathbb{R}^D \to \mathbb{R}$. Recall that the gradient of f is a (column) vector of length D whose d-th component is the derivative of $f(\boldsymbol{x})$ with respect to x_d, $\frac{\partial f(\boldsymbol{x})}{\partial x_d}$. The Hessian is the $D \times D$ matrix whose entry (i, j) is the second derivative of $f(\boldsymbol{x})$ with respect to x_i and x_j, $\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i \partial x_j}$.

Let $f : \mathbb{R}^D \to \mathbb{R}$ be the function $f(\boldsymbol{x}) = \boldsymbol{x}^\top \mathbf{A} \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x} + c$, where $\mathbf{A}$ is a (possibly asymmetric) $D \times D$ matrix, $\boldsymbol{b}$ is a vector of length D and c is a constant.

1. Determine the gradient of f, $\nabla f(\boldsymbol{x})$.
2. Determine the Hessian of f, $\nabla^2 f(\boldsymbol{x})$.

1.2 Maximum Likelihood Principle

Assume we are given i.i.d. samples $X_1, \cdots, X_N \in \mathbb{R}$ drawn from a Gaussian distribution with mean μ and variance σ^2. We do not know the two parameters (μ, σ) and want to estimate them from the data using the maximum likelihood principle.

1. Write down the likelihood for this data, i.e., the joint distribution $\mathbb{P}_{\mu,\sigma^2}(X_1, \cdots, X_2)$, where the subscripts μ and σ^2 remind us that this distribution depends on these two parameters.
2. Use the maximum likelihood principle to estimate the two parameters μ and σ^2. More precisely, find the values $\hat{\mu}(X_1, \cdots, X_N)$ and $\hat{\sigma}^2(X_1, \cdots, X_N)$ which maximize the likelihood that you computed in the previous question. *(Hint: taking the logarithm of the likelihood leads to much simpler computations)*.
3. Compute $\mathbb{E}[\hat{\mu}]$. Is this equal to the *true* parameter μ?
4. Compute $\mathbb{E}[\hat{\sigma^2}]$. Is this equal to the *true* parameter σ^2?

2 Implementing K-Means

Goals. The goal of this exercise is to

- Implement and visualize K-means clustering using the `faithful` dataset.
- Visualize the behavior with respect to the number of clusters K.
- Implement data compression using K-means.

Setup, data and sample code. Obtain the folder `labs/ex11` of the course github repository

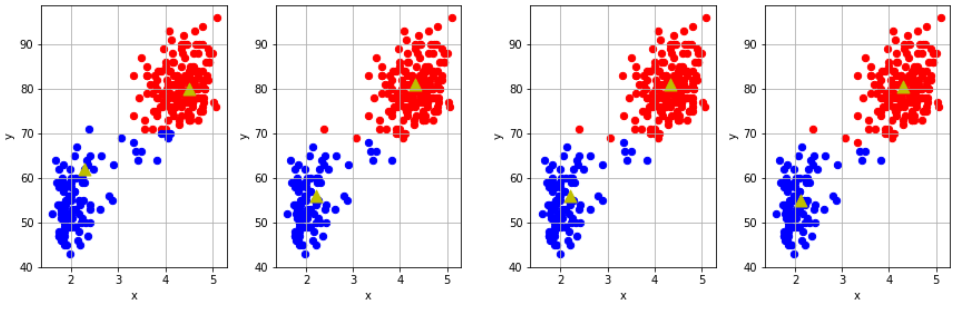
We will use the dataset `faithful.csv` in this exercise, and we have provided sample code templates that already contain useful snippets of code required for this exercise.

We will reproduce Figure 9.1 of Bishop's book.

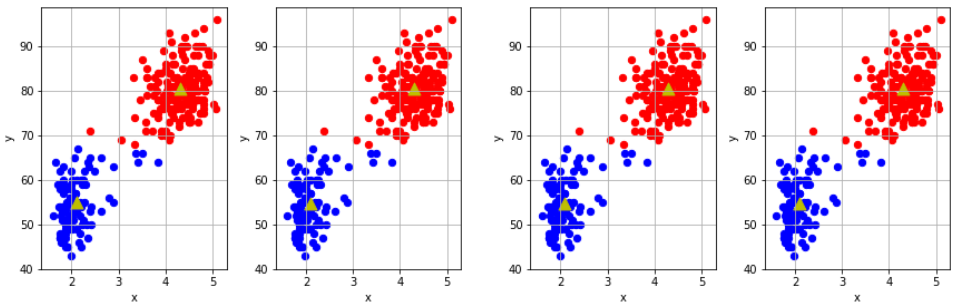
Exercise 2a):

Let's first implement K-means algorithm using the `faithful` dataset.

- Fill-in the code to initialize the cluster centers.
- Write the function `kmeansUpdate` to update the assignments z, the means μ, and the distance of data points to the means. Your code should work for any number of clusters K (not just $K = 2$).
- Write code to test for convergence.
- Visualize the output. You should get figures similar to Figure 1.



(a) Iteration 0 (b) Iteration 1



(c) Iteration 2 (d) Iteration 3

Figure 1: K-means for faithful data.

Exercise 2b):

Now, play with the initial conditions and the number of clusters to understand the behavior of K-means.

- Change the initial conditions and observe the change in convergence. The algorithm must converge for all possible initial conditions, otherwise there is a problem in your implementation.
- Try different values for K. Also try different values of initial condition. Look at the cost function value as K increases.
- BONUS: What is a good value for K? How will you choose it?

3 Data Compression using K-Means

We will implement data compression using K-means, similar to the examples shown in the class.

Exercise 3:

Write data compression for `mandrill.png`.

Your output should look like Figure 2.

Run K-means with random initializations and observe the convergence. Plot the reconstructed image by setting each pixel's value to the mean value of its cluster. Play with the number of clusters and compare the compression you get in your resulting image.

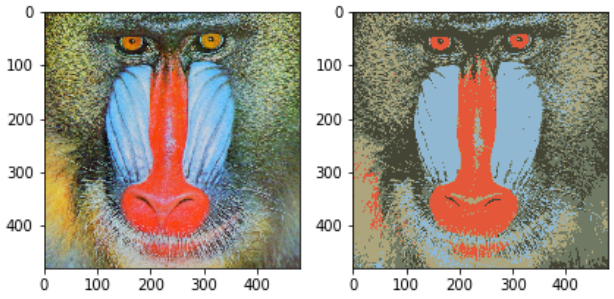


Figure 2: Image quantization / compression using K-means.