# Machine Learning
# Exercise 9

Marc Toussaint
TAs: Janik Hager, Philipp Kratzer
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

June 19, 2019

(DS BSc students may skip exercise 2, but still please read about Mixture of Gaussians and the explanations below.)

## Introduction

There is no lecture on Thursday. To still make progress, please follow this guide to learn some new material yourself. The subject is k-means clustering and Mixture of Gaussians.

**$k$-means clustering:** The method is fully described on slide 06:36 of the lecture. Again, I present the method as derived from an optimality principle. Most other references described $k$-means clustering just as a procedure. Also wikipedia https://en.wikipedia.org/wiki/K-means_clustering gives a more verbose explaination of this procedure. In my words, this is the procedure:

- We have data $D = \{x_i\}_{i=1}^n$, with $x_i \in \mathbb{R}^d$. We want to cluster the data in $K$ different clusters. $K$ is chosen ad-hoc.
- Each cluster is represented only by its mean (or center) $\mu_k \in \mathbb{R}^d$, for $k = 1, .., K$.
- We initially assign each $\mu_k$ to a random data point, $\mu_k \leftarrow x_i$ with $i \sim \mathcal{U}\{1, .., n\}$
- The algorithm also maintains an assignment mapping $c : \{1, .., n\} \to \{1, .., K\}$, which assigns each data point $x_i$ to a cluster $k = c(i)$
- For given centers $\mu_k$, we update all assignments using

$$\forall_i : \; c(i) \leftarrow \underset{c(i)}{\text{argmin}} \sum_j (x_j - \mu_{c(j)})^2 = \underset{k}{\text{argmin}} (x_i - \mu_k)^2 \;,$$

  which means we assign $x_i$ to the cluster with the nearest center $\mu_k$.
- For given assignments $c(i)$, we update all centers using

$$\forall_k : \; \mu_k \leftarrow \underset{\mu_k}{\text{argmin}} \sum_i (x_i - \mu_{c(i)})^2 = \frac{1}{|c^{-1}(k)|} \sum_{i \in c^{-1}(k)} x_i \;,$$

  that is, we set the centers equal to the mean of the data points assigned to the cluster.
- The last two steps are iterated until the assignment does not vary.

Based on this, solve exercise 1.

**Mixture of Gaussians:** Mixture of Gaussians (MoG) are very similar to $k$-means clustering. Its objective (expected likelihood maximization) is based on a probabilistic data model, which we do not go into detail here. Slide 06:40 only gives the relevant equations. A much more complete derivation of MoG as instance of Expectation Maximization is found in https://ipvs.informatik.uni-stuttgart.de/mlr/marc/teaching/15-MachineLearning/08-graphicalModels-Learning.pdf. Bishop's book https://www.microsoft.com/en-us/research/people/cmbishop/#!prml-book also gives a very good introduction. But we only need the procedural understanding here:

- We have data $D = \{x_i\}_{i=1}^n$, with $x_i \in \mathbb{R}^d$. We want to cluster the data in $K$ different clusters. $K$ is chosen ad-hoc.
- Each cluster is represented by its mean (or center) $\mu_k \in \mathbb{R}^d$ and a covariance matrix $\Sigma_k$. This covariance matrix describes an ellipsoidal shape of each cluster.

- We initially assign each $\mu_k$ to a random data point, $\mu_k \leftarrow x_i$ with $i \sim \mathcal{U}\{1,..,n\}$, and each covariance matrix to uniform (if the data is roughly uniform).
- The core difference to $k$-means: The algorithm also maintains a *probabilistic* (or soft) assignment mapping $q_i(k) \in [0,1]$, such that $\sum_{k=1}^K q_i(k) = 1$. The number $q_i(k)$ is the probability of assigning data $x_i$ to cluster $k$ (or the probability that data $x_i$ originates from cluster $k$). So, each data index $i$ is mapped to a probability over $k$, rather than a specific $k$ as in $k$-means.
- For given cluster parameters $\mu_k$, $\Sigma_k$, we update all the probabilistic assignments using

$$\forall_{i,k}:\ q_i(k) \leftarrow \mathcal{N}(x_i \,|\, \mu_k, \Sigma_k) = \frac{1}{|\, 2\pi\Sigma_k \,|^{1/2}}\ e^{-\frac{1}{2}(x_i-\mu_k)^\top\ \Sigma_k^{-1}\ (x_i-\mu_k)}$$

$$\forall_{i,k}:\ q_i(k) \leftarrow \frac{1}{\sum_{k'} q_i(k')} q_i(k)$$

  where the second line normalizes the probabilistic assignments to ensure $\sum_{k=1}^K q_i(k) = 1$.
- For given probabilistic assignments $q_i(k)$, we update all cluster parameters using

$$\forall_k:\ \mu_k \leftarrow \frac{1}{\sum_i q_i(k)}\ \sum_i q_i(k)\ x_i$$

$$\forall_k:\ \Sigma_k \leftarrow \frac{1}{\sum_i q_i(k)}\ \sum_i q_i(k)\ x_i x_i^\top - \mu_k\mu_k^\top\ ,$$

  where $\mu_k$ is the weighed mean of the data assigned to cluster $k$ (weighted with $q_i(k)$), and similarly for $\Sigma_k$.
- In this description, I skipped another parameter, $\pi_k$, which is less important and we can discuss in class.

Based on this, solve exercise 2.

Exercise 3 is extra, meaning, it's a great exercise, but beyond the default work scope.

# 1 Clustering the Yale face database (6 Points)

On the webpage find and download the Yale face database http://ipvs.informatik.uni-stuttgart.de/mlr/marc/teaching/data/yalefaces_cropBackground.tgz. The file contains gif images of 136 faces.

We'll cluster the faces using $k$-means in $K = 4$ clusters.

a) Compute a $k$-means clustering starting with random initializations of the centers. Repeat $k$-means clustering 10 times. For each run, report on the clustering error $\min \sum_i (x_i - \mu_{c(i)})^2$ and pick the best clustering. Display the center faces $\mu_k$ and perhaps some samples for each cluster. (3 P)

b) Repeat the above for various $K$ and plot the clustering error over $K$. (1 P)

c) Repeat the above on the first 20 principal components of the data. Discussion in the tutorial: Is PCA the best way to reduce dimensionality as a precursor to $k$-means clustering? What would be the 'ideal' way to reduce dimensionality as precursor to $k$-means clustering? (2 P)

# 2 Mixture of Gaussians (4 Points)

Download the data set `mixture.txt` from the course webpage, containing $n = 300$ 2-dimensional points. Load it in a data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times 2}$.

a) Implement the EM-algorithm for a Gaussian Mixture on this data set. Choose $K = 3$. Initialize by choosing the three means $\mu_k$ to be different randomly selected data points $x_i$ ($i$ random in $\{1,..,n\}$) and the covariances $\Sigma_k = \mathbf{I}$ (a more robust choice would be the covariance of the whole data). Iterate EM starting with the first E-step (computing probabilistic assignments) based on these initializations. Repeat with random restarts—how often does it converge to the optimum? (3 P)

b) Do exactly the same, but this time initialize the posterior $q_i(k)$ randomly (i.e., assign each point to a random cluster: for each point $x_i$ select $k' = rand(1:K)$ and set $q_i(k) = [k = k']$); then start EM with the first M-step. Is this better or worse than the previous way of initialization? (1 P)

# 3 Extra: Graph cut objective function & spectral clustering

One of the central messages of the whole course is: To solve (learning) problems, first formulate an objective function that defines the problem, then derive algorithms to find/approximate the optimal solution. That should also hold for clustering.

$k$-means finds centers $\mu_k$ and assignments $c : i \mapsto k$ to minimize $\min \sum_i (x_i - \mu_{c(i)})^2$.

An alternative class of objective functions for clustering are graph cuts. Consider $n$ data points with similarities $w_{ij}$, forming a weighted graph. We denote by $W = (w_{ij})$ the symmetric weight matrix, and $D = \text{diag}(d_1, .., d_n)$, with $d_i = \sum_j w_{ij}$, the degree matrix. For simplicitly we consider only 2-cuts, that is, cutting the graph in two disjoint clusters, $C_1 \cup C_2 = \{1, .., n\}$, $C_1 \cap C_2 = \emptyset$. The normalized cut objective is

$$\text{RatioCut}(C_1, C_2) = \Big(1/|C_1| + 1/|C_2|\Big) \sum_{i \in C_1, j \in C_2} w_{ij}$$

a) Let $f_i = \begin{cases} +\sqrt{|C_2|/|C_1|} & \text{for } i \in C_1 \\ -\sqrt{|C_1|/|C_2|} & \text{for } i \in C_2 \end{cases}$ be a kind of indicator function of the clustering. Prove that

$$f^\top (D - W) f = n \ \text{RatioCut}(C_1, C_2)$$

b) Further prove that $\sum_i f_i = 0$ and $\sum_i f_i^2 = n$.

Note (to be discussed in the tutorial in more detail): *Spectral clustering* addresses

$$\min f^\top (D - W) f \quad \text{s.t.} \quad \sum_i f_i = 0 \ , \quad \|f\|_2 = 1$$

by computing eigenvectors $f$ of the graph Laplacian $D - W$ with smallest eigenvalues. This is a relaxation of the above problem that minimizes over continuous functions $f \in \mathbb{R}^n$ instead of discrete clusters $C_1, C_2$. The resulting eigen functions are "approximate indicator functions of clusters". The algorithms uses $k$-means clustering in this coordinate system to explicitly decide on the clustering of data points.