SHEET #2 – SOLUTION
Due:

Fred Hamprecht
Manuel Haußmann

1 Mean-Shift and K-Means (20 pt)

In the lecture we learned two clustering approaches, whose exploration will be the task of this first exercise.

1.1 Mean-Shift (10 pt)

In the lecture, we learned about the mean-shift algorithm for clustering based on the modes of a kernel density estimate. As the Epanechnikov kernel is asymptotically optimal[1] we will rely on it for this exercise.

(a) Implementation of Epanechnikov (2pt). Implement and visualize the Epanechnikov kernel

$$k(x-\mu, w) = \frac{3}{4w}\left(1 - \left(\frac{x-\mu}{w}\right)^2\right)\mathbb{1}\left(\left|\frac{x-\mu}{w}\right| < 1\right).$$

(b) Mean-shift on a 1d data set (5pt + 3pt).

i) Implementation (5pt). As was discussed, instead of relying on a fixed step size α it is common to use an adaptive step size updating each data point directly to the local center of mass, i.e.

$$x_j^{t+1} = \frac{\sum_{i:||x_i - x_j^t||<1} x_i}{\sum_{i:||x_i - x_j^t||<1} 1}.$$

Implement this gradient ascent procedure.

ii) Visualization (3pt). Apply your Epanechnikov kernel (with $w = 1$) on `meanshift1d.npy` to perform a KDE and then the mean-shift algorithm. Visualize how the points x_j move over time, by plotting the trajectories $x_j^1, ..., x_j^T$ over time (i.e. in a 2d plot where one access is the time).

Solution: See the jupyter notebook.

1.2 K-Means (10 pt)

K-means is an algorithm that allows us to compute an unsupervised clustering of data into a fixed number of clusters. We will explore this in greater detail in this exercise.

(a) Derive the Updates (4pt). We aim to cluster a data set $\mathbf{X} \in \mathbb{R}^{p\times N}$ into K clusters, by choosing cluster centers $\mathbf{C} \in \mathbb{R}^{p\times K}$ and cluster memberships $\mathbf{M} \in [0,1]^{K\times N}$, with $\sum_k M_{kn} = 1$, such that

$$E(\mathbf{C}, \mathbf{M}; K) = ||\mathbf{X} - \mathbf{CM}||^2 = \sum_{n=1}^{N}\sum_{k=1}^{K} m_{kn}||\mathbf{x}_n - \mathbf{c}_k||^2$$

is minimized. Solve this by deriving the optimal updates for each m_{kn} and $\mathbf{c}_k$.

[1] In terms of the asymptotic mean integrated squared error, see the linked Hansen script in the lecture notes for details.

Solution: Optimize for m_{kn} by noticing that $E(\cdot)$ is linear in $\mathbf{M}$. For each datapoint $\mathbf{x}_n$

$$\sum_{k=1}^{K} m_{kn}||\mathbf{x}_n - \mathbf{c}_k||^2 \tag{1}$$

is then minimized if we set $m_{kn} = 1$ for the term with the minimal distance, i.e.

$$m_{kn} = \begin{cases} 1, & \text{if } k = \arg\min_i ||\mathbf{x}_n - \mathbf{c}_i||^2 \\ 0, & \text{otherwise} \end{cases} . \tag{2}$$

With respect to $\mathbf{c}_k$ $E(\cdot)$ is a quadratic function. Setting the derivative equal to zero give us

$$\begin{aligned} \sum_{n=1}^{N} m_{kn}(\mathbf{x}_n - \mathbf{c}_k) &= 0 \\ \Rightarrow \qquad \sum_{n=1}^{N} m_{kn}\mathbf{x}_n &= \mathbf{c}_k \sum_{n=1}^{N} m_{kn} \\ \Rightarrow \qquad \mathbf{c}_k &= \frac{\sum_{n=1}^{N} m_{kn}\mathbf{x}_n}{\sum_{n=1}^{N} m_{kn}}, \end{aligned}$$

i.e. just the average of the points assigned to the k-th cluster.

(b) Implement and apply to 2d data set (6 pt). Implement the K-means algorithms and apply it to the 2d data (`kmeans2d.npy`). Explore how the algorithm performs for different random starting values and different values of K. In each case plot how $E(\cdot)$ develops over time.

Solution: See the jupyter notebook.

2 (Bonus) K-Nearest Neighbors

With K-NN we consider the first supervised classification algorithm, i.e. additionally to the data $\mathbf{X}$ we now also have for each data point $\mathbf{x}_n$ a label $y_n \in \{1, ..., C\}$, indicating that it belongs to one of C classes. In this exercise, we will only consider a two-class problem and encode the class as $y_n \in \{0, 1\}$.

(a) Implementation. Implement the K-NN algorithm, apply it to `knn2d.npy`, and visualize the decision boundaries you get for $K \in \{1, 3, 5, 11\}$. Do this by choosing a grid of test data points classifying them according to their K neighbors.[2]

(b) Choosing K via cross-validation. As discussed in the lecture, K and other so-called hyper-parameters can often not be chosen problem-independent, but instead, need to be adapted to the problem at hand. Here, you will use cross-validation for this task. Split your data into five parts, always using the data from four parts to predict on the fifth. Calculate and plot the average prediction accuracy as a function of K for $K \in \{1, 3, 5, \ldots, 21\}$ and discuss which value of K you would choose and why.

Solution: See the jupyter notebook.

[2] Remember that you can use `numpy.meshgrid` to quickly get such a grid structure. Check the tutorial code or the documentation for details.