Deadline: Th. January 23th , 14:00 Drop your printed or legible handwritten submissions into the boxes at Samelsonplatz. Alternatively upload a `.pdf` file via LearnWeb. (e.g. exported Jupyter notebook)

1. K Means Clustering (10 points)

A. [2p] Compute the (squared) **distance matrix** $D_{ij} = \text{dist}_{\text{eucl.}}(x_i, x_j)^2$, given the data from Table 1.

x_1	x_2
0	0
0	1
-1	2
2	0
3	0
4	-1

Table 1

Solution.

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	1	5	4	9	17
x_2	1	0	2	5	10	20
x_3	5	2	0	12	20	34
x_4	4	5	12	0	1	5
x_5	9	10	20	1	0	2
x_6	17	20	34	5	2	0

B. [4p] Perform K-means clustering on the dataset from Table 1. Use the first and last datapoints as initial centers ($K = 2$). Given the final parameters, which cluster would $x^* = \binom{1}{1}$ belong to?

Solution. (Remark)

Given a dataset $X = \{x_n \mid n = 1 \dots N\}$ of datapoints in $\mathbb{R}^M$, there are two ways to choose the updated mean in K-means clustering:

① **"in-set mean":** $\hat{\mu} = \underset{\mu \in X}{\text{argmin}} \sum_n \text{dist}_{\text{eucl.}}(\mu, x_n)^2$

② **"out-of-set mean":** $\hat{\mu} = \underset{\mu \in \mathbb{R}^M}{\text{argmin}} \sum_n \text{dist}_{\text{eucl.}}(\mu, x_n)^2$

I.e. in the in-set variant, we require the center-point to always be chosen as one of the point of the actual dataset. More generally, given a dataset $X = \{x_n \mid n = 1 \dots N\}$ of datapoints in $\mathfrak{X}$ performing **K-medoids** with a given **similarity/dissimilarity measure** s / d:

	dissimilarity based	**similarity based**
"in-set medoid":	$\hat{\mu} = \underset{\mu \in X}{\text{argmin}} \sum_n d(\mu, x_n)$	$\hat{\mu} = \underset{\mu \in X}{\text{argmax}} \sum_n s(\mu, x_n)$
"out-of-set medoid":	$\hat{\mu} = \underset{\mu \in \mathfrak{X}}{\text{argmin}} \sum_n d(\mu, x_n)$	$\hat{\mu} = \underset{\mu \in \mathfrak{X}}{\text{argmax}} \sum_n s(\mu, x_n)$

Note that in this formulation K-medoids is applicable, even when the data is not vectorized (i.e. embedded in a vectorspace). For example, one could compute clusters of strings w.r.t. the LEVENSTHEIN distance (in this case, $\mathfrak{X}$ would be the set of all strings). We will now look at both the in-set and out-of-set variant.

Solution. (in-set variant)

The in-set variant has the advantage that we only need to compute the distance matrix in the beginning, and then we can use it as a lookup table for the rest of the algorithm.

iteration 1 Looking at rows of the distance matrix corresponding to the centers

	x_1	x_2	x_3	x_4	x_5	x_6
$\mu_1 = x_1$	0	1	5	4	9	17
$\mu_2 = x_6$	17	20	34	5	2	0

Then the partitions are $P_1 = \{x_1, x_2, x_3, x_4\}$ and $P_2 = \{x_5, x_6\}$. To find the new cluster centers, we sum the rows of the corresponding subtables:

	x_1	x_2	x_3	x_4	Σ
x_1	0	1	5	4	10
x_2	1	0	2	5	8
x_3	5	2	0	12	19
x_4	4	5	12	0	21

$\rightsquigarrow \mu_1' = x_2$

	x_5	x_6	Σ
x_5	0	2	2
x_6	2	0	2

$\rightsquigarrow \mu_2' = x_5$ or $\mu_2' = x_6$

We choose the former, $\mu_2' = x_5$

iteration 2 Looking at rows of the distance matrix corresponding to the centers

	x_1	x_2	x_3	x_4	x_5	x_6
$\mu_1 = x_2$	1	0	2	5	10	20
$\mu_2 = x_5$	9	10	20	1	0	2

Then the partitions are $P_1 = \{x_1, x_2, x_3\}$ and $P_2 = \{x_4, x_5, x_6\}$. To find the new cluster centers, we sum the rows of the corresponding subtables:

	x_1	x_2	x_3	Σ
x_1	0	1	5	6
x_2	1	0	2	3
x_3	5	2	0	7

$\rightsquigarrow \mu_1' = x_2$

	x_4	x_5	x_6	Σ
x_4	0	1	5	6
x_5	1	0	2	3
x_6	5	2	0	7

$\rightsquigarrow \mu_2' = x_5$

The cluster centers are the same as before, so the algorithm terminates. Finally,

$$\operatorname{dist}(\mu_1, \tbinom{1}{1})^2 = \|\tbinom{0}{1} - \tbinom{1}{1}\|^2 = 1 \quad \text{and} \quad \operatorname{dist}(\mu_2, \tbinom{1}{1})^2 = \|\tbinom{3}{0} - \tbinom{1}{1}\|^2 = 4$$

So the extra point would be considered part of cluster 1.

Solution. (out-of-set variant)

In the out-of set variant we will have to compute additional distances during the algorithm.

iteration 1 Looking at rows of the distance matrix corresponding to the centers

	x_1	x_2	x_3	x_4	x_5	x_6
$\mu_1 = x_1$	0	1	5	4	9	17
$\mu_2 = x_6$	17	20	34	5	2	0

Then the partitions are $P_1 = \{x_1, x_2, x_3, x_4\}$ and $P_2 = \{x_5, x_6\}$. To find the new cluster centers, we have to compute the means:

$$\mu_1' = \frac{1}{|P_1|} \sum_{x \in P_1} = \tfrac{1}{4}\Big(\tbinom{0}{0} + \tbinom{0}{1} + \tbinom{-1}{2} + \tbinom{2}{0}\Big) = \tfrac{1}{4}\tbinom{1}{3}$$

$$\mu_2' = \frac{1}{|P_2|} \sum_{x \in P_2} = \tfrac{1}{2}\Big(\tbinom{3}{0} + \tbinom{4}{-1}\Big) = \tfrac{1}{2}\tbinom{7}{-1}$$

iteration 2 We compute the squared euclidean distances to the new cluster centers:

	x_1	x_2	x_3	x_4	x_5	x_6
μ_1	0.625	0.125	3.125	3.625	8.125	17.125
μ_2	12.500	14.500	26.500	2.500	0.500	0.500

Then the partitions are $P_1 = \{x_1, x_2, x_3\}$ and $P_2 = \{x_4, x_5, x_6\}$. To find the new cluster centers, we have to compute the means:

$$\mu_1' = \frac{1}{|P_1|}\sum_{x\in P_1} = \tfrac{1}{3}\Big(\binom{0}{0} + \binom{0}{1} + \binom{-1}{2}\Big) = \tfrac{1}{3}\binom{-1}{3}$$
$$\mu_2' = \frac{1}{|P_2|}\sum_{x\in P_2} = \tfrac{1}{3}\Big(\binom{2}{0} + \binom{3}{0} + \binom{4}{-1}\Big) = \tfrac{1}{3}\binom{9}{-1}$$

iteration 3: We compute the squared euclidean distances to the new cluster centers:

	x_1	x_2	x_3	x_4	x_5	x_6
μ_1	1.111	0.111	1.444	6.444	12.111	22.777
μ_2	9.111	10.777	21.444	1.111	0.111	1.444

Then the partitions are $P_1 = \{x_1, x_2, x_3\}$ and $P_2 = \{x_4, x_5, x_6\}$. As these are the same as in the previous iteration, the algorithm terminates. Finally,

$$\operatorname{dist}(\mu_1, \tbinom{1}{1})^2 = \|\tfrac{1}{3}\binom{-1}{3} - \binom{1}{1}\|^2 = 16/9 \quad \text{and} \quad \operatorname{dist}(\mu_2, \tbinom{1}{1})^2 = \|\tfrac{1}{3}\binom{9}{-1} - \binom{1}{1}\|^2 = 52/9$$

So the extra point would be considered part of cluster 1.

C. [1p] For a set of points $(x_n)_{n=1:N}$ in $\mathbb{R}^m$, show that the **mean** $\hat\mu = \frac{1}{N}\sum_{n=1}^N x_n$ is the solution to the optimization problem

$$\hat\mu = \operatorname*{argmin}_{\mu\in\mathbb{R}^m} \sum_{i=1}^N \operatorname{dist}_{\text{eucl.}}(x_n, \mu)^2 \tag{1}$$

I.e. for a set of points, their mean can be characterized as the point which is, on average, closest to all the other points with respect to the **squared euclidean distance**.

Solution.

We have:

$$\hat\mu = \operatorname*{argmin}_{\mu} \sum_{in=1}^N \|\mu - x_n\|_2^2 \implies 0 \overset{!}{=} \frac{\partial}{\partial\mu}\sum_{n=1}^N \|\mu - x_n\|_2^2 = \sum_{n=1}^N 2(\mu - x_n) \implies \hat\mu = \frac{1}{N}\sum_{n=1}^N x_n$$

D⋆ [3p] For a set of points $(x_n)_{n=1:N}$ in $\mathbb{R}^m$, the **geometric median** is defined as the point

$$\hat\mu = \operatorname*{argmin}_{\mu\in\mathbb{R}^m} \sum_{n=1}^N \operatorname{dist}_{\text{eucl.}}(x_n, \mu) \tag{2}$$

Note that in contrast to the mean, (2) does not have a closed form solution. However, the minimum can be found numerically by a fixed point iteration scheme (algorithm 1). Given the dataset from Table 2 (rows are datapoints!), compute both the mean and the geometric median. What happens to both if we change the last datapoint to $\binom{100}{0}$?

x_1	x_2
-1	-1
-1	1
1	-1
1	1
10	0

Table 2

Solution.

The mean is $\mu = \binom{2}{0}$ and $\mu = \binom{20}{0}$ respectively. The geometric median however is stable with respect to the outlier and is equal to $\binom{0.6245}{0}$ in either case.

Remark (multivariate median). Note that the **geometric median** is different from the so called **marginal median** (cf. slide 10) which is defined as component wise via $\hat\mu_i = \operatorname{median}(\{X_{ni} \mid n = 1\dots N\})$. This median is easier to compute than the geometric median, but is dependent on the coordinate system. For example, in 2D if the coordinate system was rotated, i.e. instead of basis

vectors $e_1 = \binom{1}{0}$ and $e_2 = \binom{0}{1}$ we chose $\widetilde{e}_1 = \binom{\cos\theta}{\sin\theta}$ and $\widetilde{e}_2 = \binom{-\sin\theta}{\cos\theta}$, then the marginal median would change while the geometric median would stay the same.

Remark (computing the geometric median). Weizfeld's algorithm is an application of a **fixed point iteration scheme**: under certain conditions on the function f, an equation of the form $x = f(x)$ can be efficiently solved numerically by simply iteration the function. (cf. **Banach fixed-point theorem**) I.e. one starts with a random guess x_0 and then applies the function f over and over: $x_{t+1} = f(x_t)$. To see that this is whats happening, note that:

$$0 \overset{!}{=} \frac{\partial}{\partial \mu} \sum_{n=1}^{N} \|\mu - x_n\|_2 = \sum_{n=1}^{N} \frac{\mu - x_n}{\|\mu - x_n\|_2} \iff \mu = \frac{\sum_{n=1}^{N} \frac{x_n}{\|\mu - x_n\|_2}}{\sum_{n=1}^{N} \frac{1}{\|\mu - x_n\|_2}} =: f(\mu)$$

So Weizfeld's algorithm is essentially solving the first order condition for a local minimum.

Algorithm 1 Weiszfeld's algorithm

1: $\mu^{(0)} = \frac{1}{N} \sum_{n=1}^{N} x_n$
2: **for** $t = 0, 1, 2 \ldots,$ max_iter **do**
3: $\quad \mu^{(t+1)} = \left(\sum_{n=1}^{N} x_n \|x_n - \mu^{(t)}\|^{-1}\right) \Big/ \left(\sum_{n=1}^{N} \|x_n - \mu^{(t)}\|^{-1}\right)$
4: $\quad$ **if** converged **then break**

2. Gaussian Mixture Models (GMMs) (8 points)

Two datasets ("MOONS" and "STRIPES") were each clustered by 3 different methods: K-means clustering, Gaussian-Mixture-Models and Hierarchical Clustering (single link). The results are shown in Table 3.

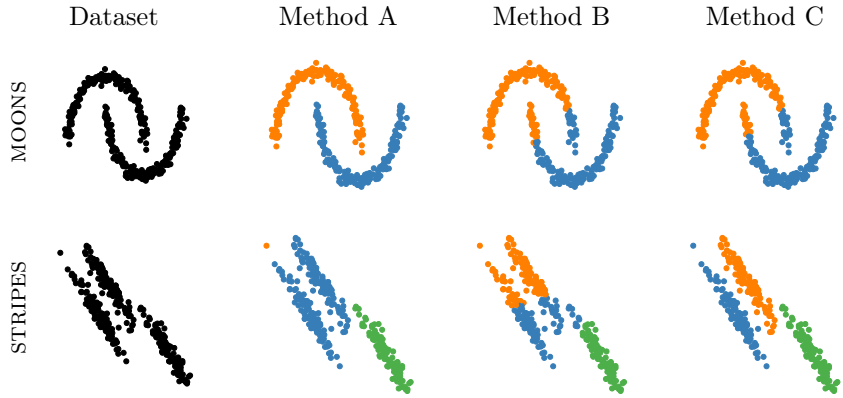

Table 3: Different Clustering Methods

A. [2p] Decide which method corresponds to A, B and C. Explain your decision.

Solution.

- Model A: must be Hierachical. Neither K-means nor GMMs could split the two moons into two seperate clusters, as the boundary between clusters in these two models is linear/quadratic.

  Another way to see it is that the mean of the one moon lies as the tip of the other. So if one whole moon was in one cluster, the tip of the other would also have to belong to that cluster in the K-means/GMM models.
- Model C: must be the GMM, due to being able to model the elliptical clusters of the stripes dataset (K-Means yields spherical clusters.)
- Model B: must be K-means as its the only one left

B. [6p] Given the data from Table 1, and the initial configuration $\pi_1, \pi_2 = \frac{1}{2}$, $\mu_1 = \binom{0}{1}$, $\mu_2 = \binom{3}{0}$, $\Sigma_1, \Sigma_2 = \mathbb{I}$, perform 1 iteration of the (soft partition) EM algorithm to fit a GMM. Which cluster would

$x^* = \binom{1}{1}$ belong to according the initial/final parameters?

Solution.

Initial soft partition:

$$q \approx \begin{bmatrix} 0.9820 & 0.9933 & 0.9999 & 0.1192 & 0.0067 & 0.0001 \\ 0.0180 & 0.0067 & 0.0001 & 0.8808 & 0.9933 & 0.9999 \end{bmatrix}$$

Updated parameters:

$$\pi_1 \approx 0.5169 \qquad \mu_1 = \begin{pmatrix} -0.2389 \\ 0.9651 \end{pmatrix} \qquad \Sigma_1 = \begin{bmatrix} 0.4391 & -0.4144 \\ -0.4144 & 0.6786 \end{bmatrix}$$

$$\pi_2 \approx 0.4831 \qquad \mu_2 = \begin{pmatrix} 3.0154 \\ -0.3425 \end{pmatrix} \qquad \Sigma_2 = \begin{bmatrix} 0.7258 & -0.3469 \\ -0.3469 & 0.2301 \end{bmatrix}$$

Final soft partition

$$q \approx \begin{bmatrix} 1.0000 & 0.9967 & 1.0000 & 0.0005 & 0.0000 & 0.0000 \\ 0.0000 & 0.0033 & 0.0000 & 0.9995 & 1.0000 & 1.0000 \end{bmatrix}$$

Cluster association of $\binom{1}{1}$:

	Cluster 1	Cluster 2
Initial:	0.8808	0.1192
Final:	0.3139	0.6861

3. Hierarchical Clustering (6 points)

A. [2p] Compute the **distance matrix** $D_{ij} = \text{dist}(x_i, x_j)$, using the **Manhatten distance** (i.e. L^1), given the data from Table 4.

x_1	x_2
0	0
1	0
2	0
-0.5	-1
0.5	-1
0	-1.5

Table 4

Solution.

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$	$\{x_6\}$
$\{x_1\}$	0	1	2	1.5	1.5	1.5
$\{x_2\}$	1	0	1	2.5	1.5	2.5
$\{x_3\}$	2	1	0	3.5	2.5	3.5
$\{x_4\}$	1.5	2.5	3.5	0	1	1
$\{x_5\}$	1.5	1.5	2.5	1	0	1
$\{x_6\}$	1.5	2.5	3.5	1	1	0

B. [4p] Perform **agglomerative Hierarchical Clustering** using **single linkage** as the cluster distance measure. Draw the associated tree (as in slides 26/27).

Solution.

We start with the individual cluster $\{x_i\}_{i=1:6}$. The lowest distance is achieved between the pairs $(\{x_1\},\{x_2\})$, $(\{x_2\},\{x_3\})$, $(\{x_4\},\{x_5\})$ and $(\{x_4\},\{x_6\})$ and $(\{x_5\},\{x_6\})$. Since we are using single linkage, we can immediately merge these to the clusters to $(\{x_1, x_2, x_3\})$ and $(\{x_4, x_5, x_6\})$. (usually, we would merge clusters one at a time, but with single linkage we can indeed merge them

at once.)

	$\{x_1\}$	$\{x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$	$\{x_6\}$
$\{x_1\}$	0	1	2	1.5	1.5	1.5
$\{x_2\}$	1	0	1	2.5	1.5	2.5
$\{x_3\}$	2	1	0	3.5	2.5	3.5
$\{x_4\}$	1.5	2.5	3.5	0	1	1
$\{x_5\}$	1.5	1.5	2.5	1	0	1
$\{x_6\}$	1.5	2.5	3.5	1	1	0

We can write an updated distance matrix table my merging the corresponding rows/cols, each time taking the minimal values (single linkage!) For example, merging $\{x_1\}$ and $\{x_2\}$ yields the updated distance table:

	$\{x_1, x_2\}$	$\{x_3\}$	$\{x_4\}$	$\{x_5\}$	$\{x_6\}$
$\{x_1, x_2\}$	0	1	1.5	1.5	1.5
$\{x_3\}$	1	0	3.5	2.5	3.5
$\{x_4\}$	1.5	3.5	0	1	1
$\{x_5\}$	1.5	2.5	1	0	1
$\{x_6\}$	1.5	3.5	1	1	0

Performing all the merges mentioned above, we arrive in the last iteration at:

	$\{x_1, x_2, x_3\}$	$\{x_4, x_5, x_6\}$
$\{x_1, x_2, x_3\}$	0	1.5
$\{x_4, x_5, x_6\}$	1.5	0

The results can be represented as a tree:

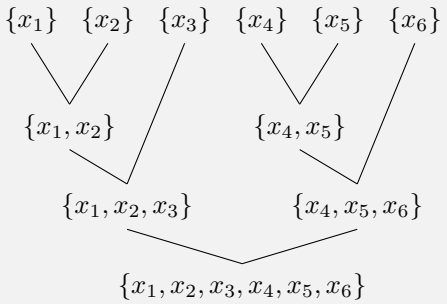

Figure 1: Hierarchical Clustering Tree (Dendogram)