Kernel for Clustering

kernel $K$-Means
Outline of Today’s Lecture

1. Review principle and steps of K-Means algorithm

2. Derive kernel version of K-means

3. Exercise: Discuss the geometrical division of the space generated by RBF versus Polynomial kernels
K-means Clustering: Iterative Technique

1. Initialization: for \( C^k, k = 1 \ldots K \) clusters, pick \( K \) arbitrary centroids and set their geometric mean to random values.

In mldemos, centroids are initialized on one datapoint with no overlap across centroids.
K-means Clustering: E-step (expectation-step)

**Assignment Step:**
- Calculate the distance from each data point to each centroid.
- Assign each data point to its “closest” centroid.

If a tie happens (i.e. two centroids are equidistant to a data point, one assigns the data point to the smallest winning centroid).

Mathematical notation:

\[ k_i = \arg \min_k \left\{ d \left( x^i, \mu^k \right) \right\} \]

- \( x^i \) \( i^{th} \) data point
- \( \mu^k \) geometric centroid
**Update step (M-Step):**
Recompute the position of centroid based on the assignment of the points
K-means Clustering: E-step (expectation-step)

\[ k_i = \arg \min_{k} \left\{ d \left( x^i, \mu^k \right) \right\} \]

- \( x^i \): \( i^{th} \) data point
- \( \mu^k \): geometric centroid

Go back to the assignment step and repeat the update step.
K-means Clustering

Stopping Criterion: stop the process when the centers are stable.
K-means Clustering

K-means creates a hard partitioning of the dataset
K-means Clustering

K-Means clustering generates a number $K$ of disjoint clusters that minimizes the quadratic cost function:

$$ J(\mu^1, ..., \mu^K) = \sum_{k=1}^{K} \sum_{x^i \in C_k} d(x^i, \mu^k) \text{ with } d(x^i, \mu^k) = \left| x^i - \mu^k \right|^2 $$
K-means Clustering: **Advantages**

- The algorithm is guaranteed to converge in a finite number of iterations (but it converges to a local optimum!)
- It is computationally **cheap** and **faster** than other clustering techniques - update step is \(\sim O(N)\).
K-means Clustering: Sensitivity

Very sensitive to the choice of the number of clusters $K$ (hyperparameter) and the initialization.
K-means Clustering: Hyperparameters

- K-means with usual norm-2 distance perform linear separation.
- Changing the power $p$ of the metric allows to generate non-linear boundaries $d(x^i, \mu^k; p) = |x^i - \mu^k|^p$
Kernel K-means

**Idea:**

- The objective function of K-means is composed of an inner product across datapoints
  - One can replace the inner product with a kernel to perform inner product in feature space.

- Exploit the principle of the kernel to perform classical K-means clustering with norm-2 in feature space:
  - This yields non-linear boundaries
  - This retains simplicity of computation of linear K-means
Kernel K-means

K – Means algorithm minimizes the objective function:

\[ J(\mu^1, \ldots, \mu^K) = \sum_{k=1}^{K} \sum_{x^j \in C^k} \|x^j - \mu^k\|^2 \]

with \( \mu^k = \frac{\sum_{x^j \in C^k} x^j}{m_k} \)

\( m_k \): number of datapoints in cluster \( C^k \)

\[ \left\{ \phi(x^i) \right\}_{i=1}^M \]

Project into a feature space

\[ J(\mu^1, \ldots, \mu^K) = \sum_{k=1}^{K} \sum_{x^j \in C^k} \left\| \phi(x^j) - \phi(\mu^k) \right\|^2 \]

\[ \frac{\sum_{x^j \in C^k} \phi(x^j)}{m_k} \]

We cannot observe the mean in feature space.

→ Construct the mean in feature space using images of points in same cluster.
Kernel K-means

\[ J(\mu^1, \ldots, \mu^K) = \sum_{k=1}^{K} \sum_{x^j \in C_k} \left( \phi(x^j) - \frac{\sum_{x^l \in C_k} \phi(x^l)}{m_k} \right)^2 \]

\[ = \sum_{k=1}^{K} \sum_{x^j \in C_k} \left( \phi(x^j)^T \phi(x^j) - \frac{2 \sum_{x^l \in C_k} \phi(x^l)^T \phi(x^j)}{m_k} + \frac{\sum_{x^j, x^l \in C_k} \phi(x^j)^T \phi(x^l)}{(m_k)^2} \right) \]

Objective function in feature space
Kernel K-means

Kernel K-means algorithm is also an iterative procedure:

1. **Initialization**: pick K clusters

2. **Assignment Step**: Assign each data point to its “closest” centroid (E-step).

\[
\arg \min_k d(x^i, C^k) = \arg \min_k \left( k(x^i, x^i) - \frac{2 \sum_{x^j \in C^k} k(x^i, x^j)}{m_k} + \sum_{x^{j'}, x^{l'} \in C^k} k(x^{j'}, x^{l'}) \right)
\]

3. **Update Step**: Update the list of points belonging to each centroid (M-step)

4. Go back to step 2 and repeat the process until the clusters are stable.
Kernel K-means: Exercise I

Metric used in kernel K-means to determine cluster assignment:

\[
\arg \min_k d\left(x^i, C^k\right) = \min_k \left\{ k\left(x^i, x^i\right) - \frac{2}{m_k} \sum_{x^j \in C^k} k\left(x^i, x^j\right) + \frac{1}{(m_k)^2} \sum_{x^j, x^{j'} \in C^k} k\left(x^j, x^{j'}\right) \right\}
\]

a) Draw the partitioning of the space when using two datapoints in 2 – dimension with a rbf kernel. Discuss the effect of the initialization.

b) Do (a) with a polynomial kernel.

Is the result affected by the placement of the datapoints?
Kernel K-means

With a RBF kernel

\[
\arg \min_k d\left(x^i, C^k\right) = \min_k \left( k\left(x^i, x^i\right) - \frac{2}{m_k} \sum_{x^j \in C^k} k\left(x^i, x^j\right) \right) + \frac{1}{m_k^2} \sum_{x^j, x^l \in C^k} k\left(x^j, x^l\right)
\]

- Cst of value 1
- If \(x^i\) is close to all points in cluster \(k\), this is close to 1.
- If the points are well grouped in cluster \(k\), this sum is close to 1.
Kernel K-means: Exercise II

Metric used in kernel K-means to determine cluster assignment:

$$\arg\min_k d\left(x^i, C^k\right) = \min_k \left( k\left(x^i, x^i\right) - \frac{2}{m_k} \sum_{x^j \in C^k} k\left(x^i, x^j\right) + \frac{\sum_{x^j, x^l \in C^k} k\left(x^j, x^l\right)}{(m_k)^2}\right)$$

i) Draw the partitioning of the space for the group of datapoints on the left when using a rbf kernel and $K = 2$.

ii) Discuss the effect of $\sigma$.

iii) Discuss the effect of the initialization.
Kernel K-means: Exercise I - solution

Top: two different runs with small \( \sigma \)
Middle: two different runs with larger \( \sigma \)
Bottom: one run with very large \( \sigma \)
\( \rightarrow \) corresponds to standard K-means
Metric used in kernel K-means to determine cluster assignment:

\[
\arg \min_k d(x^i, C^k) = \min_k \left( k(x^i, x^i) - \frac{2 \sum_{x^j \in C^k} k(x^i, x^j)}{m_k} + \frac{\sum_{x^j, x^l \in C^k} k(x^j, x^l)}{(m_k)^2} \right)
\]

a) Draw the partitioning of the space for these two clusters with a rbf kernel.

b) Discuss the effect of the number of datapoints in each cluster.
Kernel K-means: Exercise I

\[ \arg \min_k d(x^i, C^k) = \min_k \left\{ k(x^i, x^i) - \frac{2}{m_k} \sum_{x^j \in C^k} k(x^i, x^j) + \frac{\sum_{x^j, x^{l} \in C^k} k(x^j, x^{l})}{(m_k)^2} \right\} \]

The balance between numerators and denominators in A and B balances the effect of large number of datapoints and spread out clusters. A spread out cluster will gain influence over a compact cluster. Compact cluster will regain influence as the number of datapoints increases.
Kernel K-means: Exercise IV

Metric used in kernel K-means to determine cluster assignment:

$$\arg \min_k d(x^i, C^k) = \min_k \left( k(x^i, x^i) - \frac{2 \sum_{x^j \in C^k} k(x^i, x^j)}{m_k} + \frac{\sum_{x^j, x^{l} \in C^k} k(x^j, x^l)}{(m_k)^2} \right)$$

Draw the partitioning of the space when using 4 datapoints with a polynomial kernel.

Is the result affected by the placement of the datapoints?

What is the effect of the power of the polynomial $p$?
Kernel K-means

**With a polynomial kernel**

\[
\arg\min_{k} d(x^i, C^k) = \min_{k} \left( \sum_{x^j \in C^k} \sum_{x^l \in C^k} k(x^j, x^l) \right) + \sum_{x^j, x^l \in C^k} \frac{k(x^j, x^l)}{m_k} \left( m_k \right)^2
\]

A: Some of the terms change sign depending on the angle between the pair of datapoints. The relative effect of the terms depends on the position from the origin (norm).

B: If the points are aligned in the same Quadran, the sum is maximal.

A datapoint will be assigned to the cluster in the closest partition.
Result with $p=2$
Result with $p=7$: Boundary larger around the origin
Points far from the origin get outweighted by those closer to the origin.
Points far from the origin get outweighted by those closer to the origin: polynomial $p=2$.
Points far from the origin get out-weighted by those closer to the origin: polynomial $p=2$
Kernel K-means: examples

Rbf Kernel, 2 Clusters
Kernel K-means: examples

Rbf Kernel, 2 Clusters
Kernel K-means: examples
Rbf Kernel, 2 Clusters

Kernel width: 0.5

Kernel width: 0.05
Kernel K-means: Limitations

Choice of number of Clusters in Kernel K-means is important
Kernel K-means: Limitations

Choice of number of Clusters in Kernel K-means is important
Kernel K-means: Limitations

Choice of number of Clusters in Kernel K-means is important
Limitations of kernel K-means

Raw Data
Limitations of kernel K-means

kernel K-means with K=2, RBF kernel
Summary

1. Kernel K-means follows the same principle as K-means. It is an iterative procedure, akin to Expectation-Maximization.

2. As K-means, it depends on initialization of the center which is random.

3. As K-means, the solution depends on choosing well the number of clusters $K$. 