Machine Learning

CS 165B

Prof. Matthew Turk

Monday, May 16, 2016

Today

- Distance metrics and clustering (Ch. 8)

Notes

- HW#4 due on Friday (May 20)

k-Nearest neighbor (*k*NN) classifiers

- In some cases, the *k-nearest neighbor* method is preferable:
  - Classify a new instance by taking a vote of the $k \geq 1$ nearest exemplars
  - E.g., in a binary classifier, with $k = 7$, for a new input point the 7 nearest neighbors may include 5 positives and 2 negatives, so we choose positive as the classification
- Or, instead of using a fixed *k*, vote among all neighbors within a fixed radius *r*
- Or, combine the two, stopping when (*count* > *k*) or (*dist.* > *r*)
- May also use distance weighting – the closer an exemplar is to the instance, the more its vote counts (e.g., $w_i = \frac{1}{D(\boldsymbol{x}, \boldsymbol{x}_i)}$)
- What about ties?
  - Preference to the 1NN
  - Random choice
  - Etc.

Nearest neighbor classification – summary

- NN classifiers are very fast to train – $O(n)$ time
  - n = # of training samples
- But its classification is relatively slow – also $O(n)$ time
  - Need to compare the input instance with every stored training example

- Bottom line: nearest neighbor classifiers are simple, intuitive, and train quickly
  - But they can be inefficient, may require a good deal of storage, and can't easily represent a specific boundary geometry

- Importantly, NN methods rely on a useful distance metric
  - *Nearest* in Euclidian distance, Manhattan distance, Mahalanobis distance, or what?
  - This is problem-dependent
  - Distance-based methods

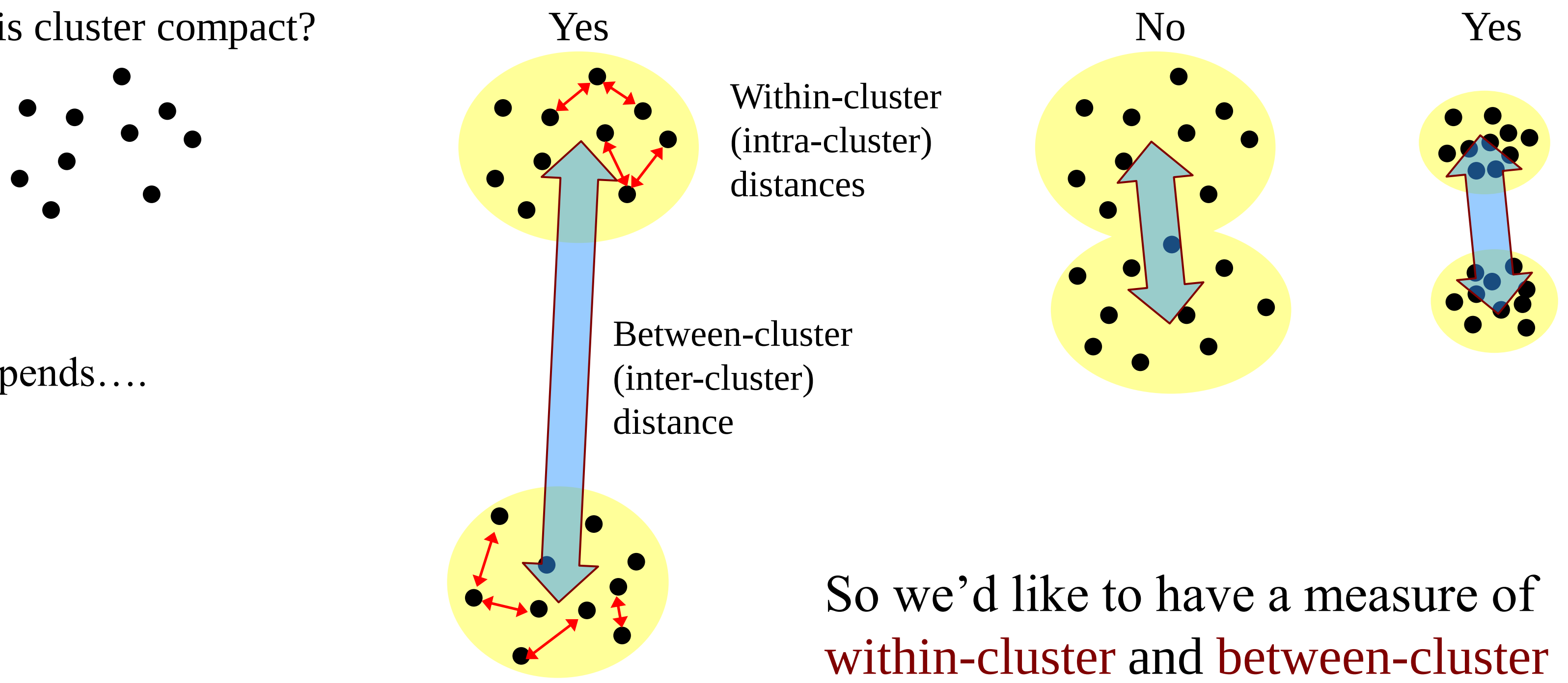
Clustering vs. classification

- Classification vs. clustering
  - In a classifier, possible class labels are provided
    - { dog, cat, elephant, mouse, …}, { spam, ham }, etc.
    - Given in the training data (for supervised classification)
  - In a clustering problem, possible labels are the cluster labels learned from the training set
    - { cluster 1, cluster 2, cluster 3, …}
    - Not given in the training data
- Terminology: In both cases, people often refer to the assigning of labels or clusters to data points (during the learning/training process, or afterwards in testing) as classification
  - Even if it's a clustering problem!

Clustering

- The goal of clustering is to find clusters (groupings) that are compact with respect to the distance metric
- What do we mean by *compactness*?

Is this cluster compact?



Yes
No
Yes
Within-cluster (intra-cluster) distances
Between-cluster (inter-cluster) distance

It depends….

So we'd like to have a measure of within-cluster and between-cluster distribution or ***scatter***

Scatter matrix

This is a widely-used concept in ML!

From the 4-25 lecture notes:

Sample covariance: $\hat{\Sigma}_{ij} = \frac{1}{k}\sum_k (x_{ik} - \hat{\mu}_i)(x_{jk} - \hat{\mu}_j) = \frac{1}{k} S_{ij}$

If X is a matrix that holds all the zero-centered samples as <u>column</u> vectors, then

$$\hat{\Sigma} = \frac{1}{k}\boxed{\boldsymbol{X_z X_z}^T} = \frac{1}{k}\boldsymbol{S}$$

S is the Scatter matrix

Alternatively, if X_z is a matrix that holds all the zero-centered samples as <u>row</u> vectors, then

$$\boldsymbol{S} = \boxed{\boldsymbol{X_z}^T \boldsymbol{X_z}}$$

It depends on how we define $\boldsymbol{X_z}$ **!**

For the scatter matrix (and thus the covariance matrix), $\boldsymbol{X_z}$ is zero-mean

– That is, the mean data point $\bar{\boldsymbol{x}}$ (or $\boldsymbol{\mu}$ or $\boldsymbol{\mu_x}$) is first subtracted from every data point $\boldsymbol{x}_i$

By the way, the Gram matrix is not zero-mean…

Scatter matrix

- If the data D is partitioned into K subsets $\{D_1, D_2, \ldots D_K\}$ then the scatter matrix can be written as

$$\boldsymbol{S} = \left(\sum_{j=1}^{K} \underset{\uparrow}{\boldsymbol{S}_j}\right) + \boldsymbol{B} \leftarrow$$

Scatter matrix of the partition means

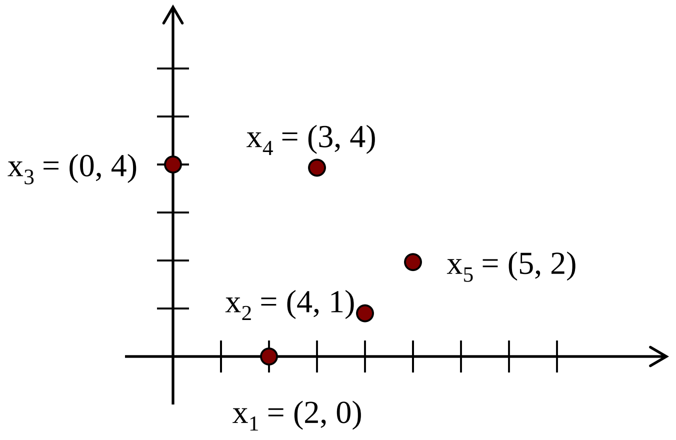
Between-cluster scatter matrix

Subset scatter matrices

Within-cluster scatter matrices

To compute B, replace every point in D with the mean of its partition D_i and compute the scatter matrix

Example: Scatter matrix



$$\boldsymbol{X} = [\boldsymbol{x}_1 \quad \boldsymbol{x}_2 \quad \boldsymbol{x}_3 \quad \boldsymbol{x}_4 \quad \boldsymbol{x}_5] = \begin{bmatrix} 2 & 4 & 0 & 3 & 5 \\ 0 & 1 & 4 & 4 & 2 \end{bmatrix}$$

The Gram matrix is …

$$\boldsymbol{G} = \boldsymbol{X}^T\boldsymbol{X} = \begin{bmatrix} 4 & 8 & 0 & 6 & 10 \\ 8 & 17 & 4 & 16 & 22 \\ 0 & 4 & 16 & 16 & 8 \\ 6 & 16 & 16 & 25 & 23 \\ 10 & 22 & 8 & 23 & 29 \end{bmatrix}$$

($k \times k$), where k is the number of data points

$$\overline{\boldsymbol{x}} = \frac{1}{5}\sum_{i=1}^{5} \boldsymbol{x}_i = \frac{1}{5}\begin{bmatrix} 14 \\ 11 \end{bmatrix} = \begin{bmatrix} 2.8 \\ 2.2 \end{bmatrix}$$

$$\boldsymbol{X}_z = [\boldsymbol{x}_1 - \overline{\boldsymbol{x}} \quad \boldsymbol{x}_2 - \overline{\boldsymbol{x}} \quad \boldsymbol{x}_3 - \overline{\boldsymbol{x}} \quad \boldsymbol{x}_4 - \overline{\boldsymbol{x}} \quad \boldsymbol{x}_5 - \overline{\boldsymbol{x}}] = \begin{bmatrix} -0.8 & 1.2 & -2.8 & 0.2 & 2.2 \\ -2.2 & -1.2 & 1.8 & 1.8 & -0.2 \end{bmatrix}$$

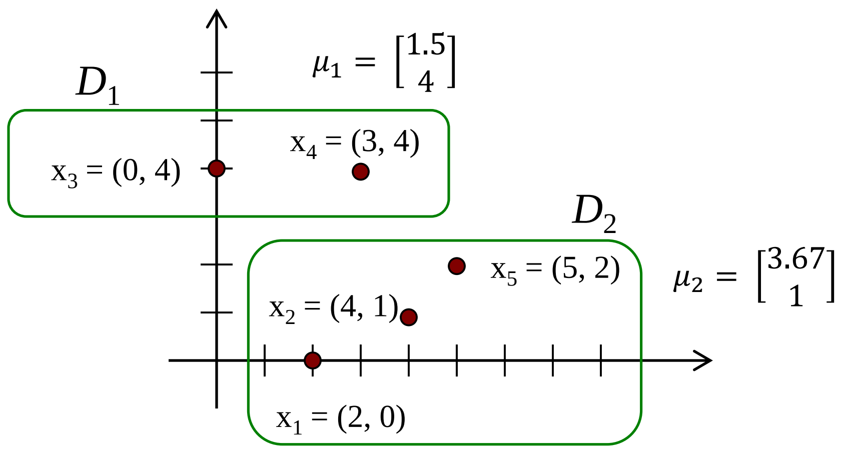
The Scatter matrix is …

$$\boldsymbol{S} = \boldsymbol{X}_z{\boldsymbol{X}_z}^T = \begin{bmatrix} 14.8 & -4.8 \\ -4.8 & 12.8 \end{bmatrix}$$

(N x N), where N is the dimensionality of the data points

Example: Scatter matrix via partitions

$$\boldsymbol{S} = \left(\sum_{j=1}^{K} \boldsymbol{S}_j\right) + \boldsymbol{B}$$



D_1
$\mu_1 = \begin{bmatrix}1.5\\4\end{bmatrix}$
$x_3 = (0, 4)$
$x_4 = (3, 4)$
D_2
$x_5 = (5, 2)$
$x_2 = (4, 1)$
$x_1 = (2, 0)$
$\mu_2 = \begin{bmatrix}3.67\\1\end{bmatrix}$

Partition means

$$\begin{bmatrix}1.5 & 1.5 & 3.67 & 3.67 & 3.67\\ 4 & 4 & 1 & 1 & 1\end{bmatrix} \qquad \mu_B = \begin{bmatrix}2.8\\2.2\end{bmatrix}$$

Zero-mean partition means

$$B_z = \begin{bmatrix}-1.3 & -1.3 & 13/15 & 13/15 & 13/15\\ 1.8 & 1.8 & -1.2 & -1.2 & -1.2\end{bmatrix}$$

Between-cluster scatter matrix

$$\boldsymbol{B} = \boldsymbol{B}_z\boldsymbol{B}_z{}^T = \begin{bmatrix}5.633 & -7.8\\ -7.8 & 10.8\end{bmatrix}$$

Scatter matrix of D_1

$$\boldsymbol{S}_1 = \left[\boldsymbol{x}_3 - \begin{bmatrix}1.5\\4\end{bmatrix} \quad \boldsymbol{x}_4 - \begin{bmatrix}1.5\\4\end{bmatrix}\right]\left[\boldsymbol{x}_3 - \begin{bmatrix}1.5\\4\end{bmatrix} \quad \boldsymbol{x}_4 - \begin{bmatrix}1.5\\4\end{bmatrix}\right]^T = \begin{bmatrix}-1.5 & 1.5\\ 0 & 0\end{bmatrix}\begin{bmatrix}-1.5 & 1.5\\ 0 & 0\end{bmatrix}^T = \begin{bmatrix}4.5 & 0\\ 0 & 0\end{bmatrix}$$

Scatter matrix of D_2

$$\boldsymbol{S}_2 = \begin{bmatrix}-5/3 & 1/3 & 4/3\\ -1 & 0 & 1\end{bmatrix}\begin{bmatrix}-5/3 & 1/3 & 4/3\\ -1 & 0 & 1\end{bmatrix}^T = \begin{bmatrix}4.67 & 3\\ 3 & 2\end{bmatrix}$$

$$\boldsymbol{S} = \boldsymbol{S}_1 + \boldsymbol{S}_1 + \boldsymbol{B} = \begin{bmatrix}4.5 & 0\\ 0 & 0\end{bmatrix} + \begin{bmatrix}4.67 & 3\\ 3 & 2\end{bmatrix} + \begin{bmatrix}5.633 & -7.8\\ -7.8 & 10.8\end{bmatrix} = \begin{bmatrix}14.8 & -4.8\\ -4.8 & 12.8\end{bmatrix}$$

Scatter

- The scatter of X is defined as the trace of the scatter matrix
  - The *trace* is the sum of the diagonal elements of a square matrix

$$\mathrm{Scat}(\boldsymbol{X}) = Tr(\boldsymbol{S}) = Tr(\boldsymbol{X}_z{\boldsymbol{X}_z}^T)$$

$$= Tr\left(\begin{bmatrix} 14.8 & -4.8 \\ -4.8 & 12.8 \end{bmatrix}\right) = 14.8 + 12.8 = 27.6$$

- Since ***S*** can be decomposed into partitions, so can Scat(***X***)

$$\mathrm{Scat}(D) = \underbrace{\sum_{j=1}^{K} \mathrm{Scat}(D_j)}_{} + \underbrace{\sum_{j=1}^{K} |D_j| \, ||\boldsymbol{\mu}_j - \boldsymbol{\mu}||^2}_{}$$

Fixed for a given data set

Want to choose partitions that minimize this

Equivalent to maximizing this

This is the goal of *k*-means clustering

K-means clustering

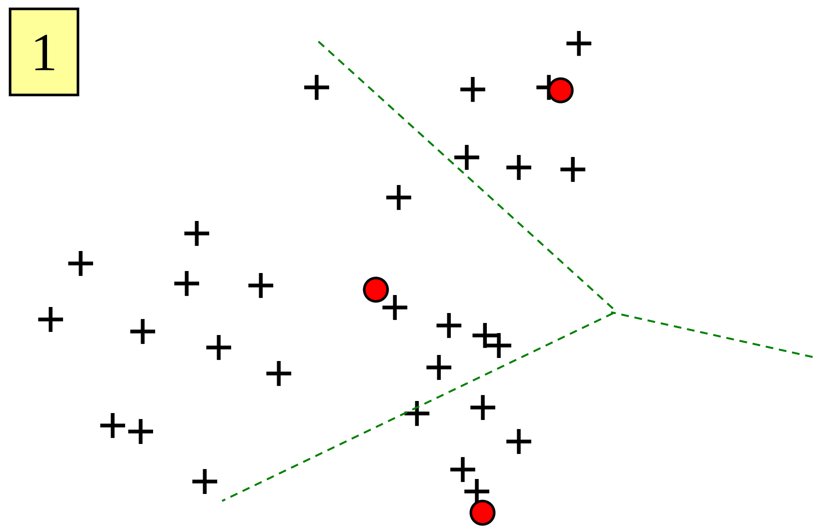
- The general K-means clustering problem is NP-complete, so there is no efficient solution to find the optimal clustering (data partition)
- A widely-used heuristic algorithm for clustering is also known as the K-means algorithm, but it is not optimal
  - It will converge to a solution, but there is no guarantee that the solution is the best one (the global minimum of scatter)
  - But it works quite well in most cases!
- Typically, the K-means algorithm would be run several times (with a random starting point) and then the best solution is selected
  - I.e., the solution with the smallest within-cluster scatter

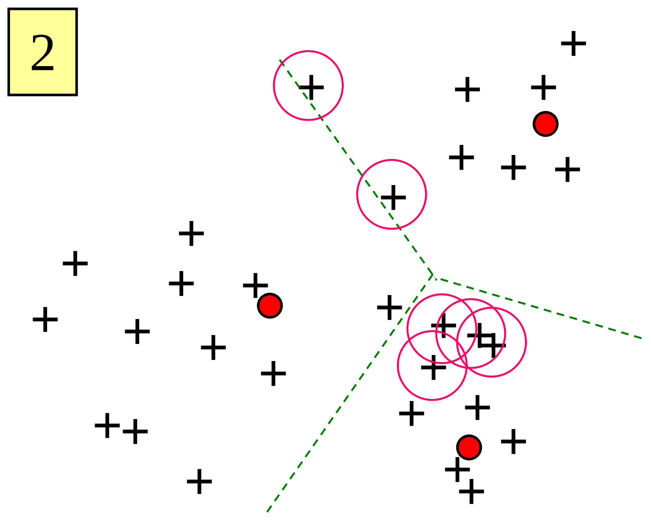
K-means algorithm

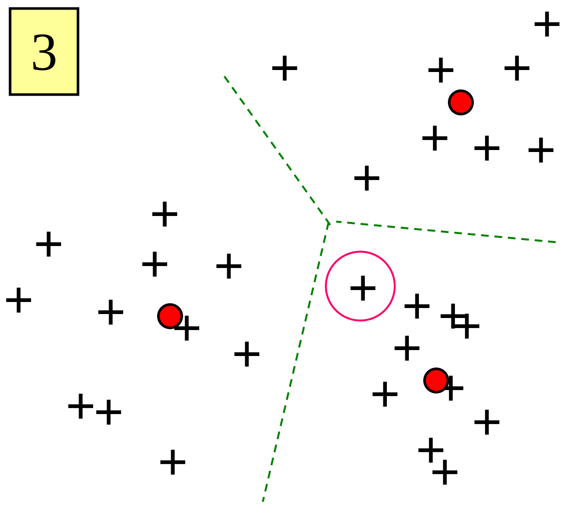
K is an input parameter

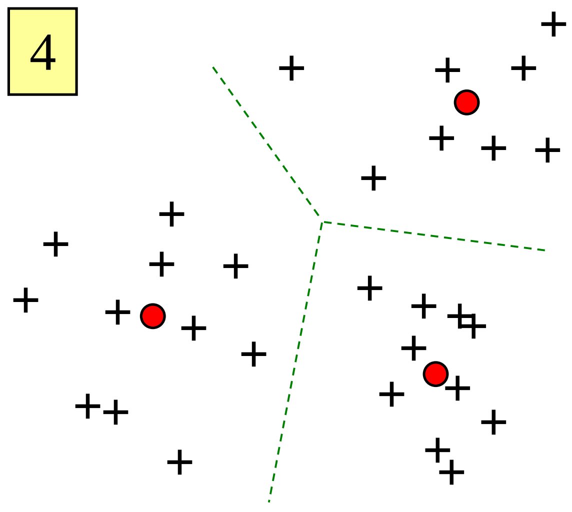
Algorithm KMeans(D, K) – K-means clustering using Euclidean distance Dis_2.

```
Input  : data $D \subseteq \mathbb{R}^d$; number of clusters $K \in \mathbb{N}$.
Output : $K$ cluster means $\mu_1, \dots, \mu_K \in \mathbb{R}^d$.
randomly initialise $K$ vectors $\mu_1, \dots, \mu_K \in \mathbb{R}^d$;
repeat
    assign each $\mathbf{x} \in D$ to $\arg\min_j \mathrm{Dis}_2(\mathbf{x}, \mu_j)$;   ⟵ 1-Nearest neighbor assignment
    for $j = 1$ to $K$ do
        $D_j \leftarrow \{\mathbf{x} \in D | \mathbf{x}$ assigned to cluster $j\}$;   ⟵ Partition defined by assignment
        $\mu_j = \frac{1}{|D_j|} \sum_{\mathbf{x} \in D_j} \mathbf{x}$;   ⟵ Re-compute the cluster mean
    end
until no change in $\mu_1, \dots, \mu_K$;
return $\mu_1, \dots, \mu_K$;
```


1


2


3

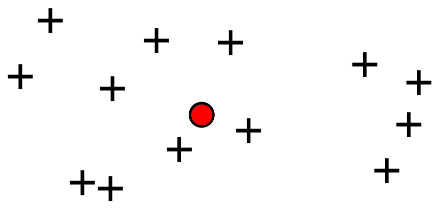

4

K-medoids algorithm

- In some problems, the cluster exemplars (representatives) are required to be data points
  - As opposed to using the mean of the cluster points, for example, since the mean is most likely not a point in the data set
- The concept of *medoid* is useful here – the medoid of a set of points is the point with the minimal average dissimilarity (distance) to all other points in the set
  - Using some distance metric: Euclidian, L1, etc.
  - This is a generalization of the concept of median to multiple dimensions
- K-means can be modified to use data points as exemplars rather than means

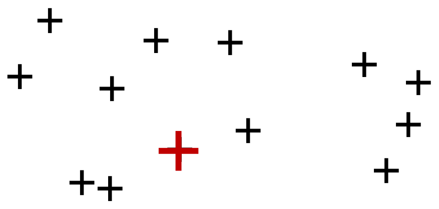
K-medoids algorithm

Cluster mean



Location that minimizes the sum of squared distances to points

Cluster medoid



Point that minimizes the sum of squared distances to points

K-medoids algorithm

Algorithm KMedoids(D, K, Dis) – K-medoids clustering using arbitrary distance metric Dis.

```
Input  : data $D \subseteq \mathscr{X}$; number of clusters $K \in \mathbb{N}$;
         distance metric $\mathrm{Dis} : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$.
Output : $K$ medoids $\mu_1, \ldots, \mu_K \in D$, representing a predictive clustering of $\mathscr{X}$.
randomly pick $K$ data points $\mu_1, \ldots, \mu_K \in D$;
repeat
    assign each $\mathbf{x} \in D$ to $\arg\min_j \mathrm{Dis}(\mathbf{x}, \mu_j)$;
    for $j = 1$ to $K$ do
        $D_j \leftarrow \{\mathbf{x} \in D | \mathbf{x}$ assigned to cluster $j\}$;
        $\mu_j = \arg\min_{\mathbf{x} \in D_j} \sum_{\mathbf{x}' \in D_j} \mathrm{Dis}(\mathbf{x}, \mathbf{x}')$;   $\longleftarrow$ Re-compute the cluster medoid
    end
until no change in $\mu_1, \ldots, \mu_K$;
return $\mu_1, \ldots, \mu_K$;
```

Kernel K-means clustering

Algorithm Kernel-KMeans(D, K) – K-means clustering using kernelised distance Dis_κ.

```
Input   : data $D \subseteq \mathscr{X}$; number of clusters $K \in \mathbb{N}$.
Output  : $\underline{K\text{-fold partition } D_1 \uplus \ldots \uplus D_K = D}$.
randomly initialise $K$ clusters $D_1, \ldots, D_K$;
repeat
    assign each $\mathbf{x} \in D$ to $\arg\min_j \frac{1}{|D_j|} \sum_{\mathbf{y} \in D_j} \underline{\text{Dis}_\kappa(\mathbf{x}, \mathbf{y})}$;   ⟵ Re-assign each point to a partition according to the minimum average (kernel) distance
    for $j = 1$ to $K$ do
        $D_j \leftarrow \{\mathbf{x} \in D | \mathbf{x}$ assigned to cluster $j\}$;
    end
until no change in $D_1, \ldots, D_K$;
return $D_1, \ldots, D_K$;
```

As before, replace the dot product with a kernel function $\boldsymbol{\kappa}$

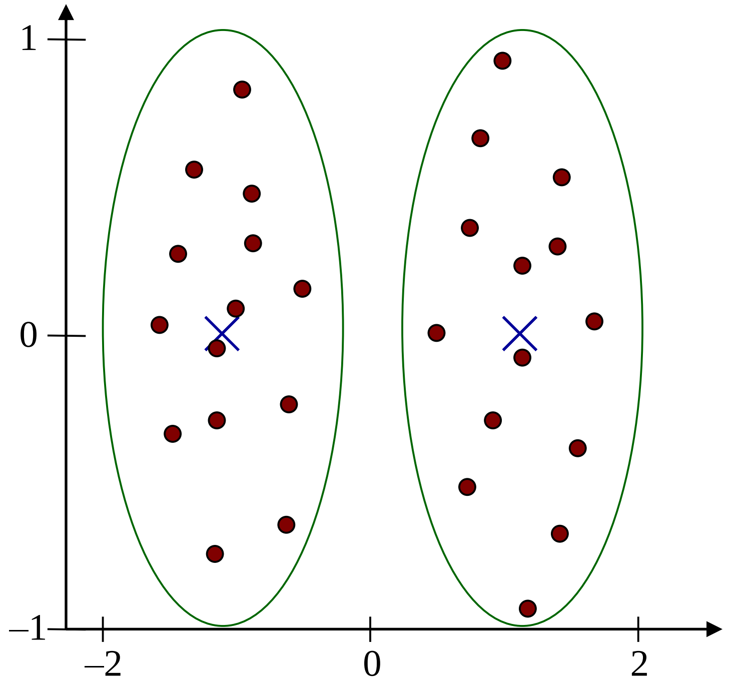
$$\text{Dis}_2(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||_2 = \sqrt{(\mathbf{x} - \mathbf{y}) \cdot (\mathbf{x} - \mathbf{y})} = \sqrt{\mathbf{x} \cdot \mathbf{x} - 2\mathbf{x} \cdot \mathbf{y} + \mathbf{y} \cdot \mathbf{y}}$$

$$\text{Dis}_\kappa(\mathbf{x}, \mathbf{y}) = \sqrt{\kappa(\mathbf{x}, \mathbf{x}) - 2\kappa(\mathbf{x}, \mathbf{y}) + \kappa(\mathbf{y}, \mathbf{y})}$$

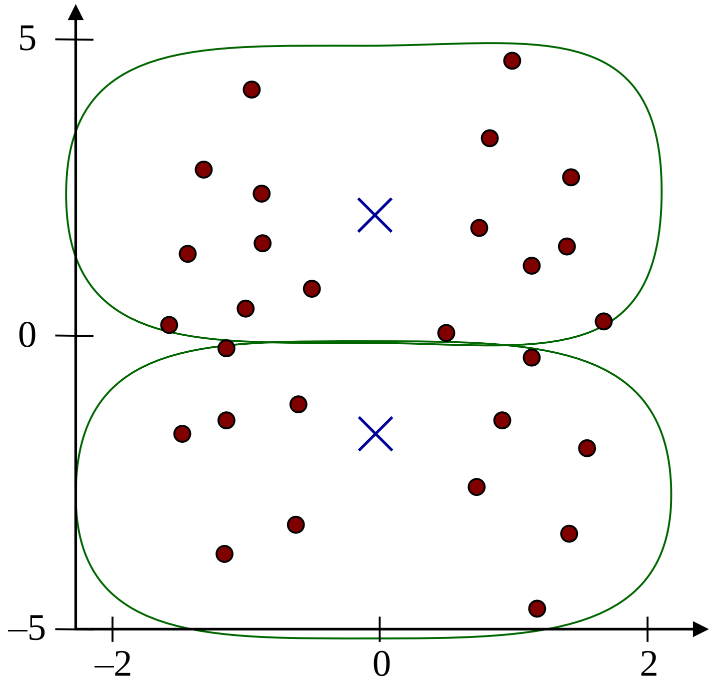
Clustering

- These clustering methods are distance-based methods that do not take into account information about the cluster shape
  - This can lead to counter-intuitive and unwanted results

K = 2



K-means finds the two natural clusters



After scaling the data in *y*, the found clusters are not what we expected!

Summary: Distance methods and clustering

- Euclidian distance may not always be the right choice
- Similarity is a function of distance
- Nearest neighbor methods assign classes/clusters based on distances to points or exemplars, not based on computed boundaries
- For good clustering, we want high within-class (intra-class) similarity and low between-class (inter-class) similarity
- The scatter matrix is an important structure in clustering
- The K-means algorithm (and variations) is widely used