M5MS10
Machine Learning

Spring 2018

Lecture 9

Dr Ben Calderhead

b.calderhead@imperial.ac.uk

M5MS10
Machine Learning

Spring 2018

Lecture 9

Dr Ben Calderhead

b.calderhead@imperial.ac.uk

IS10
Learning

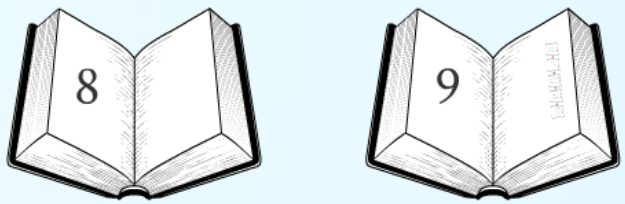
2018

re 9

lderhead

mperial.ac.uk

Clustering and Density Estimation



Cluster Analysis

Unsupervised Learning

Cluster analysis allows us to discover subgroups of patterns whose members are more similar to each other than they are to members of other groups. These clusters

- can provide clear interpretable meaning in many applications,
- can be used to *label* the data and train classifiers,
- can be used to detect departures from some expected characteristics of the data.

This is in contrast to dimensionality reduction, the other form of unsupervised learning that we considered previously, which allows us to discover low-dimensional projections that

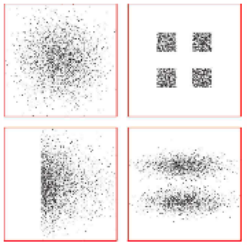
- can be used to *visualise* the data,
- represent *features* or *input vectors* that replace the original variables and can be used to more efficiently train a classifier on labelled data.

Example

Suppose the data is generated from a *normal distribution* with mean vector $\mathbf{m}$ and covariance matrix Σ.

From a geometric point of view, we can consider the samples $\{\mathbf{x}_i\}$ as a cloud of points in a p dimensional space.

Example



Modelling Assumptions

When the Gaussian model provides an accurate description of the data, the two *sufficient statistics* summarise the data well.

- $\mathbf{m}$ locates the *centre of gravity* of the data cloud and best represents the data in the sense of minimising the sum of squared distances from $\mathbf{m}$ to each sample.
- Σ describes the amount of spread there is in the data in each dimension around $\mathbf{m}$.

However, if the normality *assumption* does not hold, the sufficient statistics will give a misleading description of the data!

Cluster Analysis

Cluster analysis aims to identify coherent structures in the data, and raises the following questions:

How is the coherence of the groupings to measured?

How are coherent groupings to be identified?

We shall consider a simple algorithm for this, *K-means clustering*, and see the direct connection this has with the EM algorithm that we considered in the previous lecture.

Cluster Analysis

There are two main approaches to clustering.

1. Fit probabilistic models that capture the features of the data.
   - Can be difficult to pursue if little is known about the *data generating mechanism*.
   - Non-parametric density estimation techniques can be used, but gets harder in higher dimensions without lots and lots of data.
2. Design algorithms that detect groups of patterns.
   - Define a measure of *similarity* between every pair of data points, and find partitions of the data such that data in the same group are more *similar* compared to data points in other groups.
   - Define meaningful objective functions that capture the clusters.

Unsupervised Learning

Cluster analysis allows us to discover subgroups of patterns whose members are more similar to each other than they are to members of other groups. These clusters

- can provide clear interpretable meaning in many applications,
- can be used to *label* the data and train classifiers,
- can be used to detect departures from some expected characteristics of the data.

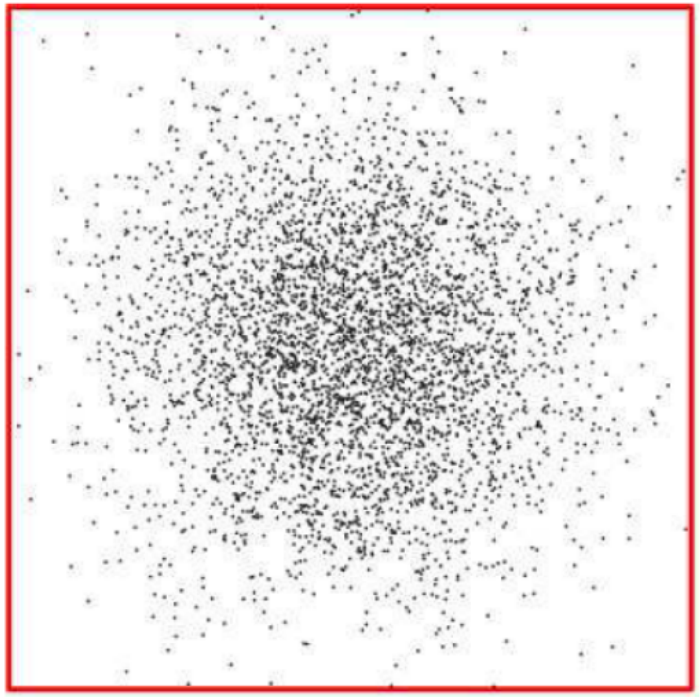
This is in contrast to dimensionality reduction, the other form of unsupervised learning that we considered previously, which allows us to discover low-dimensional projections that

- can be used to *visualise* the data,
- represent *features* or *input vectors* that replace the original variables and can be used to more efficiently train a classifier on labelled data.

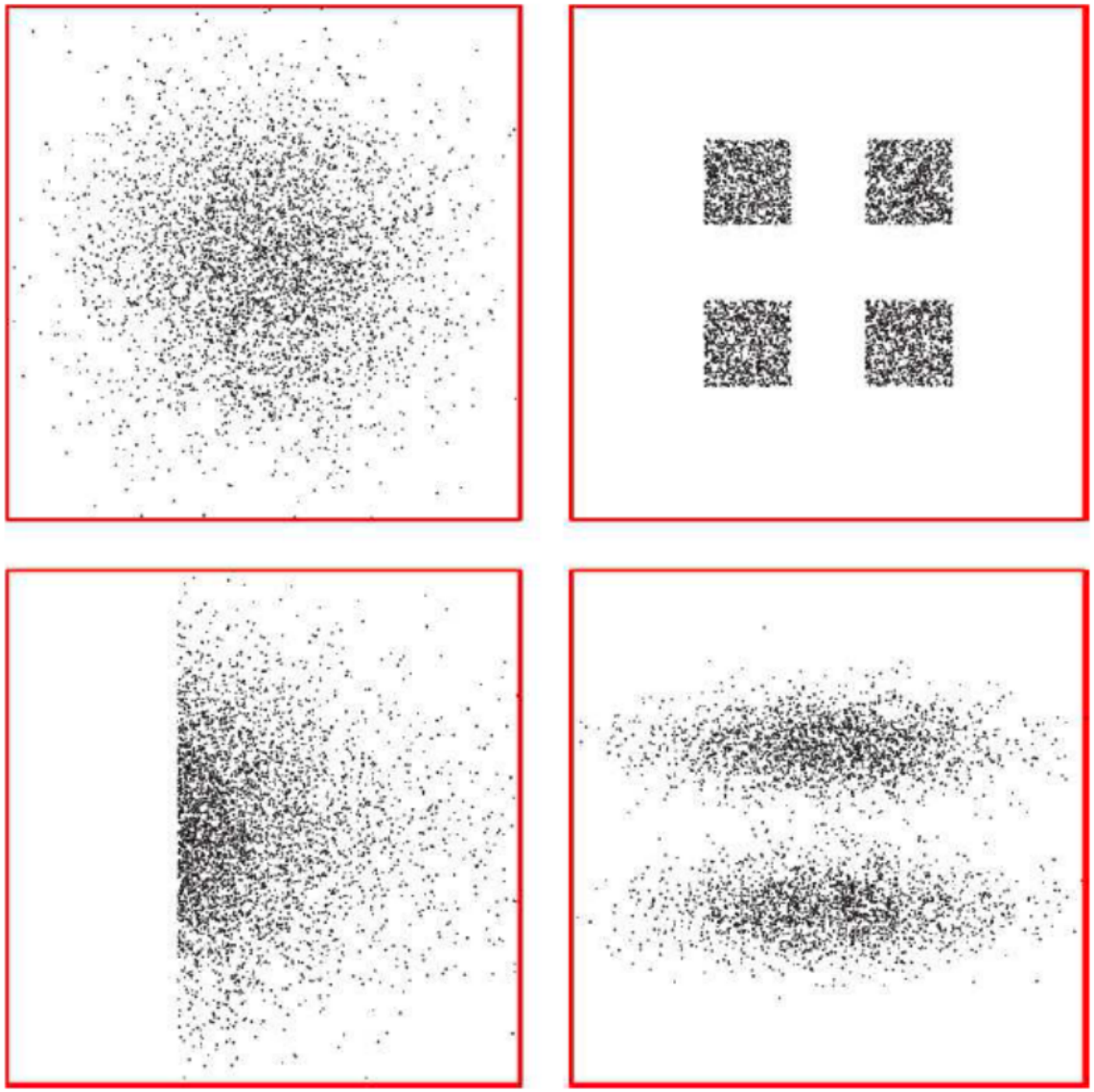
Example

Suppose the data is generated from a *normal distribution* with mean vector $\mathbf{m}$ and covariance matrix Σ.

From a geometric point of view, we can consider the samples $\{\mathbf{x}_i\}$ as a cloud of points in a p dimensional space.



Example



Modelling Assumptions

When the Gaussian model provides an accurate description of the data, the two *sufficient statistics* summarise the data well.

- $\mathbf{m}$ locates the *centre of gravity* of the data cloud and best represents the data in the sense of minimising the sum of squared distances from $\mathbf{m}$ to each sample.
- Σ describes the amount of spread there is in the data in each dimension around $\mathbf{m}$.

However, if the normality *assumption* does not hold, the sufficient statistics will give a misleading description of the data!

Cluster Analysis

Cluster analysis aims to identify coherent structures in the data, and raises the following questions:

How is the coherence of the groupings to measured?

How are coherent groupings to be identified?

We shall consider a simple algorithm for this, *K-means clustering*, and see the direct connection this has with the EM algorithm that we considered in the previous lecture.

Cluster Analysis

There are two main approaches to clustering.

1. Fit probabilistic models that capture the features of the data.
   - Can be difficult to pursue if little is known about the *data generating mechanism*.
   - Non-parametric density estimation techniques can be used, but gets harder in higher dimensions without lots and lots of data.
2. Design algorithms that detect groups of patterns.
   - Define a measure of *similarity* between every pair of data points, and find partitions of the data such that data in the same group are more *similar* compared to data points in other groups.
   - Define meaningful objective functions that capture the clusters.

Cluster Analysis

Unsupervised Learning

Cluster analysis allows us to discover subgroups of patterns whose members are more similar to each other than they are to members of other groups. These clusters

- can provide clear interpretable meaning in many applications,
- can be used to *label* the data and train classifiers,
- can be used to detect departures from some expected characteristics of the data.

This is in contrast to dimensionality reduction, the other form of unsupervised learning that we considered previously, which allows us to discover low-dimensional projections that

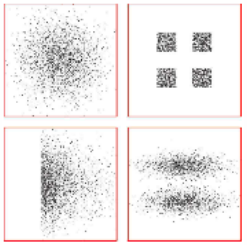
- can be used to *visualise* the data,
- represent *features* or *input vectors* that replace the original variables and can be used to more efficiently train a classifier on labelled data.

Example

Suppose the data is generated from a *normal distribution* with mean vector $\mathbf{m}$ and covariance matrix Σ.

From a geometric point of view, we can consider the samples $\{\mathbf{x}_i\}$ as a cloud of points in a p dimensional space.

Example



Modelling Assumptions

When the Gaussian model provides an accurate description of the data, the two *sufficient statistics* summarise the data well.

- $\mathbf{m}$ locates the *centre of gravity* of the data cloud and best represents the data in the sense of minimising the sum of squared distances from $\mathbf{m}$ to each sample.
- Σ describes the amount of spread there is in the data in each dimension around $\mathbf{m}$.

However, if the normality *assumption* does not hold, the sufficient statistics will give a misleading description of the data!

Cluster Analysis

Cluster analysis aims to identify coherent structures in the data, and raises the following questions:

How is the coherence of the groupings to measured?

How are coherent groupings to be identified?

We shall consider a simple algorithm for this, *K-means clustering*, and see the direct connection this has with the EM algorithm that we considered in the previous lecture.

Cluster Analysis

There are two main approaches to clustering.

1. Fit probabilistic models that capture the features of the data.
   - Can be difficult to pursue if little is known about the *data generating mechanism*.
   - Non-parametric density estimation techniques can be used, but gets harder in higher dimensions without lots and lots of data.
2. Design algorithms that detect groups of patterns.
   - Define a measure of *similarity* between every pair of data points, and find partitions of the data such that data in the same group are more *similar* compared to data points in other groups.
   - Define meaningful objective functions that capture the clusters.

K-Means Clustering

K-Means Algorithm

We shall assume a collection of N data points $\mathbf{x}_n \in \mathbb{R}^D$ in some D dimensional space.

Assume at most K possible groupings or *clusters*, with a *binary indicator* associated with each data point and cluster $z_{kn} \in \{0, 1\}$.

We shall see similarities with density estimation, however in this context our algorithm is *less complex* since we don't need a distribution over the indicator variables.

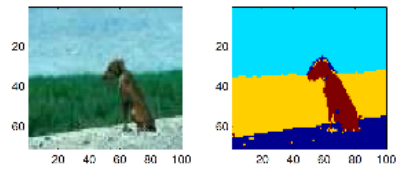
K-Means Illustration

As an example let's consider a picture of a "wee dog" looking out to sea.

The image is a 100×100 colour JPEG, and we can represent each pixel in the image as corresponding to the Red, Green and Blue channels of the image.

The aim is to segment the image into self-consistent regions, corresponding to the background or foreground, using clusters of pixels based on their distances in the 3 dimensional Red, Green, Blue space.

K-Means Illustration



K-Means Algorithm

We shall assume a collection of N data points $\mathbf{x}_n \in \mathbb{R}^D$ in some D dimensional space.

Assume at most K possible groupings or *clusters*, with a *binary indicator* associated with each data point and cluster $z_{kn} \in \{0, 1\}$.

We shall see similarities with density estimation, however in this context our algorithm is *less complex* since we don't need a distribution over the indicator variables.

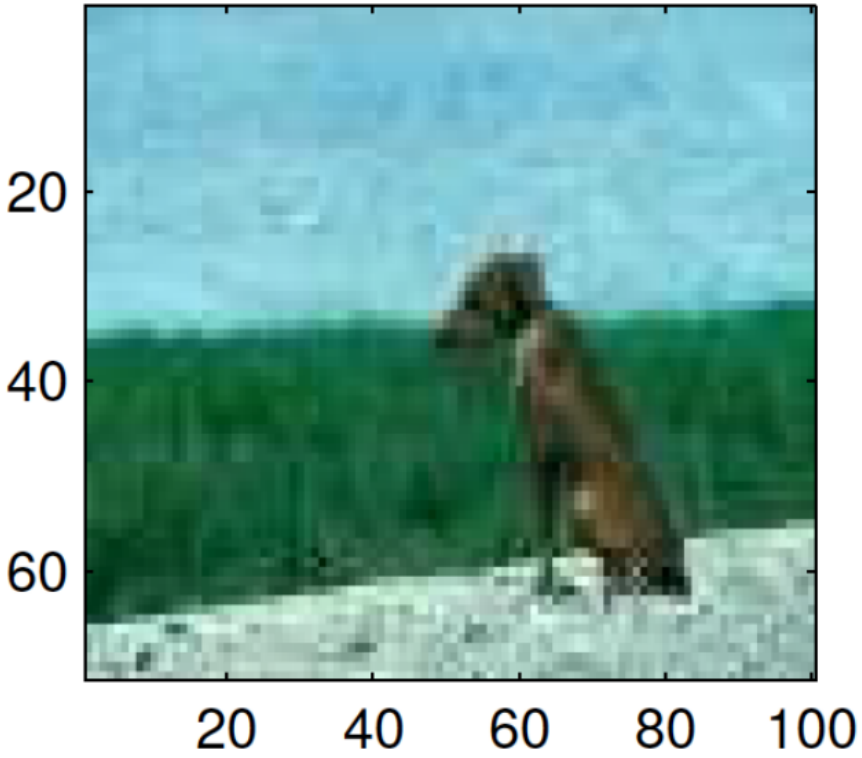
K-Means Illustration

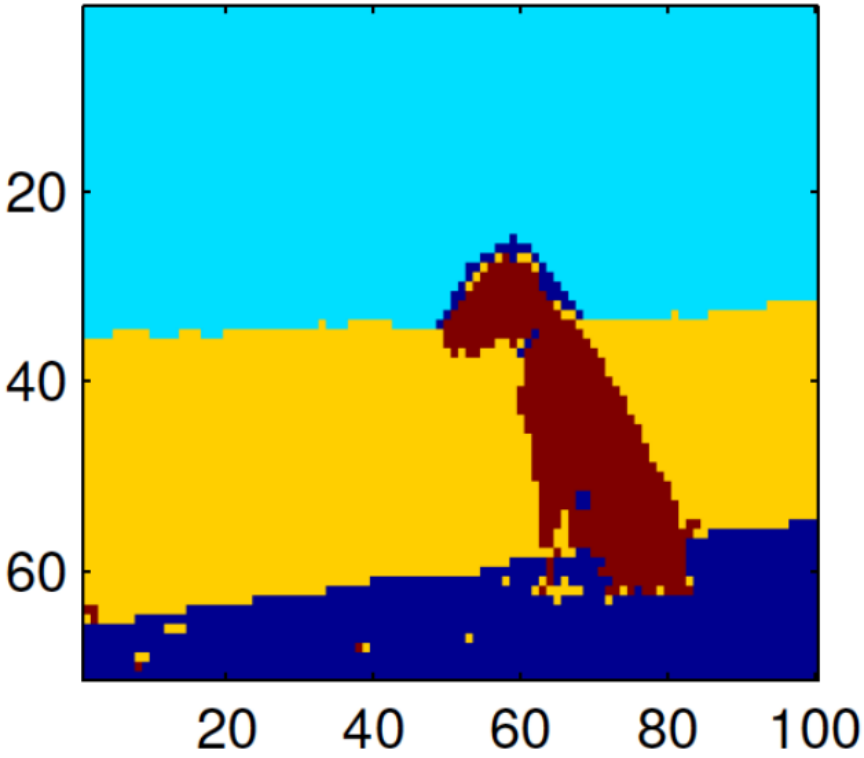
As an example let's consider a picture of a "wee dog" looking out to sea.

The image is a 100×100 colour JPEG, and we can represent each pixel in the image as corresponding to the Red, Green and Blue channels of the image.

The aim is to segment the image into self-consistent regions, corresponding to the background or foreground, using clusters of pixels based on their distances in the 3 dimensional Red, Green, Blue space.

K-Means Illustration





K-Means Clustering

K-Means Algorithm

We shall assume a collection of N data points $\mathbf{x}_n \in \mathbb{R}^D$ in some D dimensional space.

Assume at most K possible groupings or *clusters*, with a *binary indicator* associated with each data point and cluster $z_{kn} \in \{0, 1\}$.

We shall see similarities with density estimation, however in this context our algorithm is *less complex* since we don't need a distribution over the indicator variables.

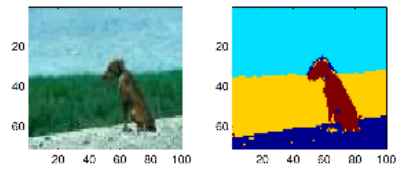
K-Means Illustration

As an example let's consider a picture of a "wee dog" looking out to sea.

The image is a 100×100 colour JPEG, and we can represent each pixel in the image as corresponding to the Red, Green and Blue channels of the image.

The aim is to segment the image into self-consistent regions, corresponding to the background or foreground, using clusters of pixels based on their distances in the 3 dimensional Red, Green, Blue space.

K-Means Illustration



K-Means Algorithm

K-Means

We now need a measure of the *cohesiveness* of the points allocated to each cluster, which we can define in terms of distances to the cluster average.

We can define a measure of cluster compactness as the total distance from the cluster mean,

$$\sum_{\mathbf{x}_n \in C_k} ||\mathbf{x}_n - \mathbf{m}_k||^2 = \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

where the cluster mean is defined as

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{\mathbf{x}_n \in C_k} \mathbf{x}_n$$

and $N_k = \sum_{n=1}^{N} z_{kn}$ is the total number of points allocated to the cluster k.

K-Means

The total goodness of the clustering will then be based on the sum of the cluster compactness measures for each of the K clusters. Using the indicator variables z_{kn}, we can then define the overall cluster goodness as

$$\mathcal{E}_K = \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

Now that we have an overall measure of cluster quality, the next step is to devise an algorithm that will allow us to optimise this.

K-Means

Two sets of parameters to optimise, the cluster mean values $\mathbf{m}_k$ and the cluster allocation indicator variables z_{kn}.

We shall optimise our criterion over each set of variables by alternately holding one set fixed - similar to the EM algorithm!

Given the current z_{kn}, the optimal value of the mean vectors $\mathbf{m}_k$ are simply the estimates based on the data points allocated to each cluster, and so we obtain our K-mean by,

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} z_{kn} \mathbf{x}_{kn}}{\sum_{n'=1}^{N} z_{kn'}}$$

K-Means

Now given each of our new values of $\mathbf{m}_k$, we need to update the values of our indicator values z_{kn}.

From the expression for $\mathcal{E}_K$, we can see that each $\mathbf{x}_n$ should be assigned to the cluster k for which it has the shortest distance to the cluster centre.

In other words, $||x_n - \mathbf{m}_k||^2$ is the smallest for all values of $k = 1, \ldots, K$.

Therefore, we set $z_{kn} = 1$ for the k that yields the minimum of $||x_n - \mathbf{m}_k||^2$.

K-Means

Once these values have been redefined then we can again revise our estimates of each $\mathbf{m}_k$ and continue this iteration until $\mathcal{E}_k$ converges to some steady value.

Another Image Example

The objective in the image analysis application is to partition the image into regions, each of which has a reasonably homogenous visual appearance.

After clustering, each data point $\mathbf{x}_n$ is replaced by its cluster mean $\mathbf{m}_k$.

For a given K, the algorithm therefore represents the image using only K colours.

An obvious limitation of this is that the *spatial ordering* of the pixels is not taken into account.

Another Image Example

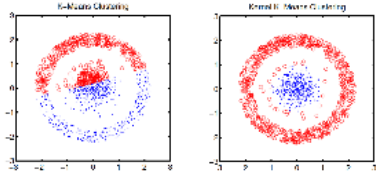
K-Means Shortcomings

The converged solution will vary with the initial conditions and the algorithm relies on a value of K being supplied by the user.

As we shall see, K-means relies on splitting the feature space using linear hyper-planes.

If there exist nonlinear relationships between the features, then standard K-means will fail.

Example



K-Means

We now need a measure of the *cohesiveness* of the points allocated to each cluster, which we can define in terms of distances to the cluster average.

We can define a measure of cluster compactness as the total distance from the cluster mean,

$$\sum_{\mathbf{x}_n \in \mathcal{C}_k} ||\mathbf{x}_n - \mathbf{m}_k||^2 = \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

where the cluster mean is defined as

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{\mathbf{x}_n \in \mathcal{C}_k} \mathbf{x}_n$$

and $N_k = \sum_{n=1}^{N} z_{kn}$ is the total number of points allocated to the cluster k.

K-Means

The total goodness of the clustering will then be based on the sum of the cluster compactness measures for each of the K clusters. Using the indicator variables z_{kn}, we can then define the overall cluster goodness as

$$\mathcal{E}_K = \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

Now that we have an overall measure of cluster quality, the next step is to devise an algorithm that will allow us to optimise this.

K-Means

Two sets of parameters to optimise, the cluster mean values $\mathbf{m}_k$ and the cluster allocation indicator variables z_{kn}.

We shall optimise our criterion over each set of variables by alternately holding one set fixed - similar to the EM algorithm!

Given the current z_{kn}, the optimal value of the mean vectors $\mathbf{m}_k$ are simply the estimates based on the data points allocated to each cluster, and so we obtain our K-mean by,

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} z_{kn}\mathbf{x}_{kn}}{\sum_{n'=1}^{N} z_{kn'}}$$

K-Means

Now given each of our new values of $\mathbf{m}_k$, we need to update the values of our indicator values z_{kn}.

From the expression for $\mathcal{E}_K$, we can see that each $\mathbf{x}_n$ should be assigned to the cluster k for which it has the shortest distance to the cluster centre.

In other words, $||x_n - \mathbf{m}_k||^2$ is the smallest for all values of $k = 1, \ldots, K$.

Therefore, we set $z_{kn} = 1$ for the k that yields the minimum of $||x_n - \mathbf{m}_k||^2$.

K-Means

Once these values have been redefined then we can again revise our estimates of each $\mathbf{m}_k$ and continue this iteration until $\mathcal{E}_k$ converges to some steady value.

Another Image Example

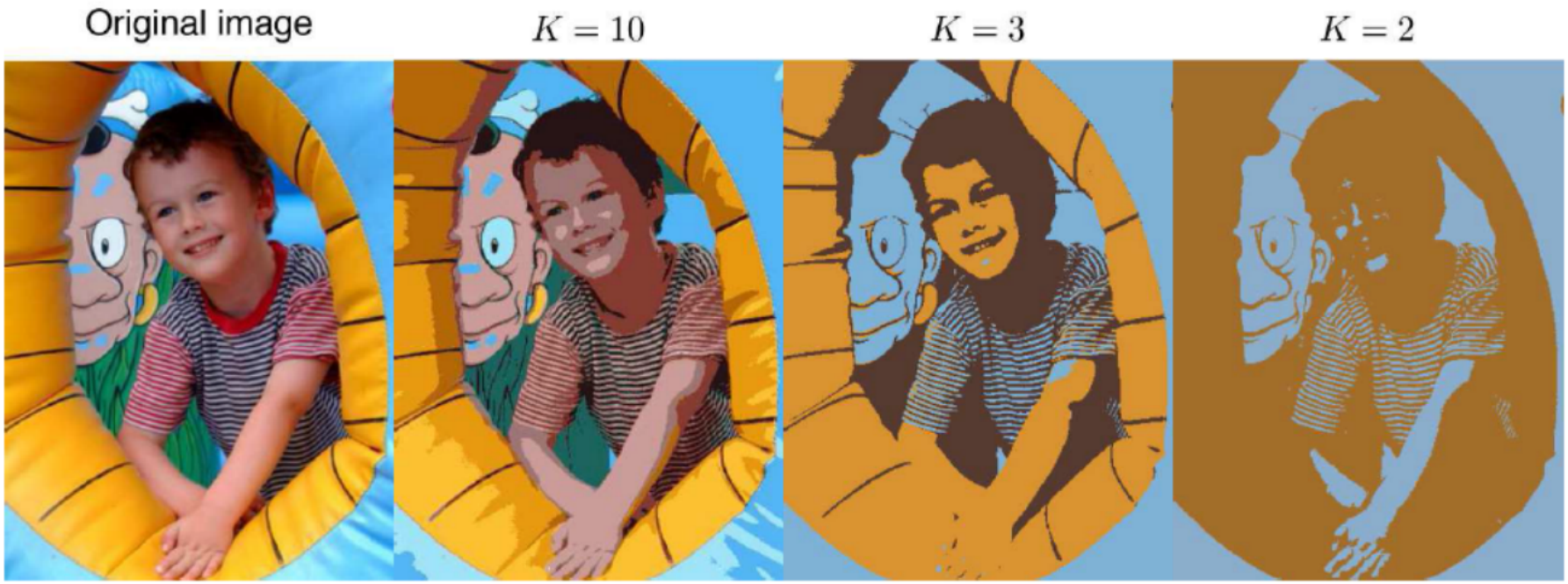
The objective in the image analysis application is to partition the image into regions, each of which has a reasonably homogenous visual appearance.

After clustering, each data point $\mathbf{x}_n$ is replaced by its cluster mean $\mathbf{m}_k$.

For a given K, the algorithm therefore represents the image using only K colours.

An obvious limitation of this is that the *spatial ordering* of the pixels is not taken into account.

Another Image Example



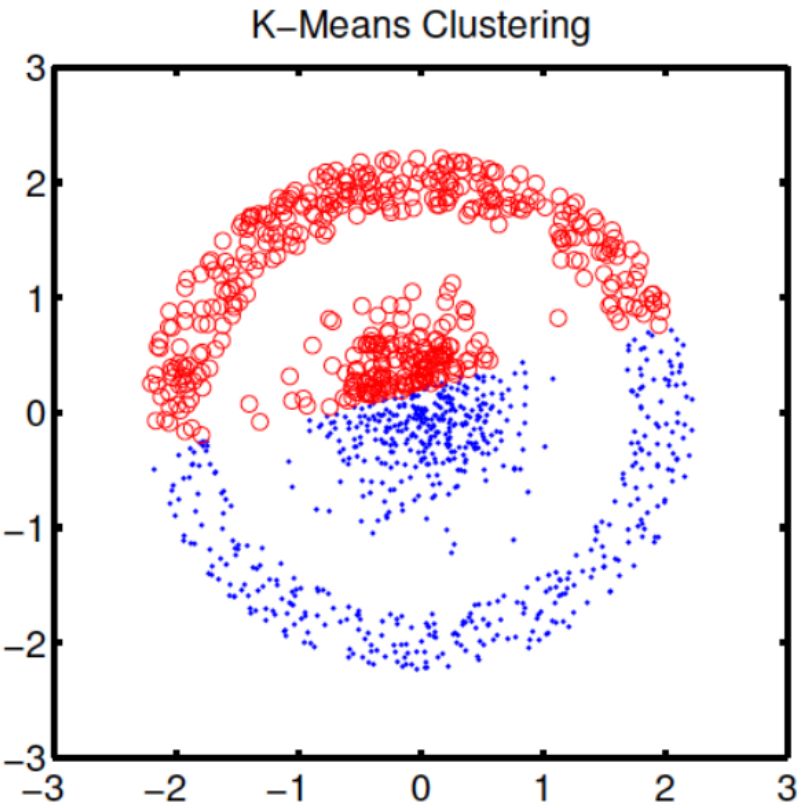
K-Means Shortcomings

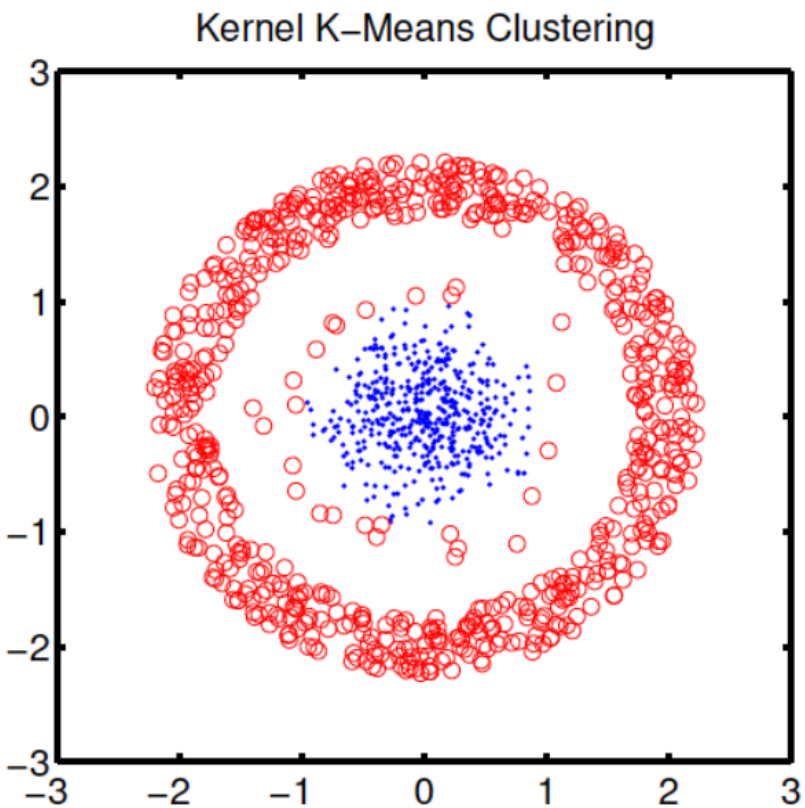
The converged solution will vary with the initial conditions and the algorithm relies on a value of K being supplied by the user.

As we shall see, K-means relies on splitting the feature space using linear hyper-planes.

If there exist nonlinear relationships between the features, then standard K-means will fail.

Example


K–Means Clustering
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3


Kernel K–Means Clustering
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3

K-Means Algorithm

K-Means

We now need a measure of the *cohesiveness* of the points allocated to each cluster, which we can define in terms of distances to the cluster average.

We can define a measure of cluster compactness as the total distance from the cluster mean,

$$\sum_{\mathbf{x}_n \in C_k} ||\mathbf{x}_n - \mathbf{m}_k||^2 = \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

where the cluster mean is defined as

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{\mathbf{x}_n \in C_k} \mathbf{x}_n$$

and $N_k = \sum_{n=1}^{N} z_{kn}$ is the total number of points allocated to the cluster k.

K-Means

The total goodness of the clustering will then be based on the sum of the cluster compactness measures for each of the K clusters. Using the indicator variables z_{kn}, we can then define the overall cluster goodness as

$$\mathcal{E}_K = \sum_{k=1}^{K} \sum_{n=1}^{N} z_{kn} ||\mathbf{x}_n - \mathbf{m}_k||^2$$

Now that we have an overall measure of cluster quality, the next step is to devise an algorithm that will allow us to optimise this.

K-Means

Two sets of parameters to optimise, the cluster mean values $\mathbf{m}_k$ and the cluster allocation indicator variables z_{kn}.

We shall optimise our criterion over each set of variables by alternately holding one set fixed - similar to the EM algorithm!

Given the current z_{kn}, the optimal value of the mean vectors $\mathbf{m}_k$ are simply the estimates based on the data points allocated to each cluster, and so we obtain our K-mean by,

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} z_{kn} \mathbf{x}_{kn}}{\sum_{n'=1}^{N} z_{kn'}}$$

K-Means

Now given each of our new values of $\mathbf{m}_k$, we need to update the values of our indicator values z_{kn}.

From the expression for $\mathcal{E}_K$, we can see that each $\mathbf{x}_n$ should be assigned to the cluster k for which it has the shortest distance to the cluster centre.

In other words, $||x_n - \mathbf{m}_k||^2$ is the smallest for all values of $k = 1, \ldots, K$.

Therefore, we set $z_{kn} = 1$ for the k that yields the minimum of $||x_n - \mathbf{m}_k||^2$.

K-Means

Once these values have been redefined then we can again revise our estimates of each $\mathbf{m}_k$ and continue this iteration until $\mathcal{E}_k$ converges to some steady value.

Another Image Example

The objective in the image analysis application is to partition the image into regions, each of which has a reasonably homogenous visual appearance.

After clustering, each data point $\mathbf{x}_n$ is replaced by its cluster mean $\mathbf{m}_k$.

For a given K, the algorithm therefore represents the image using only K colours.

An obvious limitation of this is that the *spatial ordering* of the pixels is not taken into account.

Another Image Example

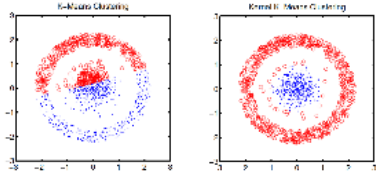
K-Means Shortcomings

The converged solution will vary with the initial conditions and the algorithm relies on a value of K being supplied by the user.

As we shall see, K-means relies on splitting the feature space using linear hyper-planes.

If there exist nonlinear relationships between the features, then standard K-means will fail.

Example



Kernel K-Means

Kernel K-Means

Let's consider the clustering criterion that is used in K-means,

$$\begin{aligned}\mathcal{E}_K &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\mathbf{x}_n - \mathbf{m}_k||^2 \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n - \mathbf{m}_k)^T(\mathbf{x}_n - \mathbf{m}_k) \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n^T\mathbf{x}_n - 2\mathbf{m}_k^T\mathbf{x}_n + \mathbf{m}_k^T\mathbf{m}_k)\end{aligned}$$

and in particular we can look at the terms $\mathbf{m}_k^T\mathbf{x}_n$ and $\mathbf{m}_k^T\mathbf{m}_k$.

Kernel K-Means

We can easily see that

$$\mathbf{m}_k^T\mathbf{x}_n = \frac{1}{N_k}\sum_{m=1}^{N} z_{km}\mathbf{x}_m^T\mathbf{x}_n$$

and

$$\begin{aligned}\mathbf{m}_k^T\mathbf{m}_k &= \left(\frac{1}{N_k}\sum_{p=1}^{N} z_{kp}\mathbf{x}_p\right)^2 \\ &= \frac{1}{N_k^2}\sum_{p=1}^{N}\sum_{l=1}^{N} z_{kp}z_{kl}\mathbf{x}_p^T\mathbf{x}_l\end{aligned}$$

We therefore see that the objective function can be defined purely in terms of *inner products* of pairs of data points.

Kernel K-Means

We can therefore define a more general kernel K-means algorithm that takes into account more flexible representations of the nonlinear relationships that might be present in the data.

$$\begin{aligned}\mathcal{E}_K^\phi &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\phi(\mathbf{x}_n) - \mathbf{m}_k^\phi||^2 \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(K(\mathbf{x}_n, \mathbf{x}_n)\right) \\ &\quad - \sum_{k=1}^{K}\sum_{m=1}^{N} z_{km}\left(\frac{2}{N_k}\sum_{m=1}^{N} z_{km}K(\mathbf{x}_m, \mathbf{x}_n)\right) \\ &\quad - \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(\frac{1}{N_k^2}\sum_{p=1}^{N}\sum_{l=1}^{N} z_{kp}z_{kl}K(\mathbf{x}_p, \mathbf{x}_l)\right)\end{aligned}$$

We must also worry about the appropriate choice of parameters.

EM and K-Means

In the previous lecture, we developed an EM algorithm for a Gaussian Mixture model - we can have another look at this algorithm and make some simplifying assumptions.

We can assume that the covariance for each mixture component is simply a fixed identity matrix.

There is therefore no need to worry about estimating these, if they are fixed.

We can also think about the posterior probabilities of the data points being allocated to a Gaussian component.

EM and K-Means

In this simplified case, where each Σ_k is a fixed identity matrix, then the E-step is simply

$$E\{z_{kn}\} = p(k|\mathbf{x}_n) \propto \exp\left(-\frac{1}{2}||\mathbf{x}_n - \mathbf{m}_k||^2\right)$$

and the M-step is just

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} p(k|\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} p(k|\mathbf{x}_n)}$$

EM and K-Means

If we make a hard decision about the expected value of z_{kn} based on the maximum of the posterior, we should be able to see that the maximum of the posterior corresponds to the minimum of $||\mathbf{x}_n - \mathbf{m}_k||^2$. This is exactly what we are doing in K-means!

If we choose z_{kn} based on the maximum posterior, our M-step is exactly the cluster centre updates for K-means clustering.

K-means clustering can therefore be obtained directly from the EM algorithm using a mixture of unit variance isotropic Gaussians, where at the E-step a hard decision about component membership is made, rather than assigning a probability of membership.

Kernel K-Means

Let's consider the clustering criterion that is used in K-means,

$$
\begin{aligned}
\mathcal{E}_K &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\mathbf{x}_n - \mathbf{m}_k||^2 \\
&= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n - \mathbf{m}_k)^T(\mathbf{x}_n - \mathbf{m}_k) \\
&= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n^T\mathbf{x}_n - 2\mathbf{m}_k^T\mathbf{x}_n + \mathbf{m}_k^T\mathbf{m}_k)
\end{aligned}
$$

and in particular we can look at the terms $\mathbf{m}_k^T\mathbf{x}_n$ and $\mathbf{m}_k^T\mathbf{m}_k$.

Kernel K-Means

We can easily see that

$$\mathbf{m}_k^T \mathbf{x}_n = \frac{1}{N_k} \sum_{m=1}^{N} z_{km} \mathbf{x}_m^T \mathbf{x}_n$$

and

$$\begin{aligned} \mathbf{m}_k^T \mathbf{m}_k &= \left(\frac{1}{N_k} \sum_{p=1}^{N} z_{kp} \mathbf{x}_p \right)^2 \\ &= \frac{1}{N_k^2} \sum_{p=1}^{N} \sum_{l=1}^{N} z_{kp} z_{kl} \mathbf{x}_p^T \mathbf{x}_l \end{aligned}$$

We therefore see that the objective function can be defined purely in terms of *inner products* of pairs of data points.

Kernel K-Means

We can therefore define a more general kernel K-means algorithm that takes into account more flexible representations of the nonlinear relationships that might be present in the data.

$$
\begin{aligned}
\mathcal{E}_K^{\phi} &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\phi(\mathbf{x}_n) - \mathbf{m}_k^{\phi}||^2 \\
&= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(K(\mathbf{x}_n,\mathbf{x}_n)\right) \\
&\quad - \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(\frac{2}{N_k}\sum_{m=1}^{N} z_{km}K(\mathbf{x}_m,\mathbf{x}_n)\right) \\
&\quad - \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(\frac{1}{N_k^2}\sum_{p=1}^{N}\sum_{l=1}^{N} z_{kp}z_{kl}K(\mathbf{x}_p,\mathbf{x}_l)\right)
\end{aligned}
$$

We must also worry about the appropriate choice of parameters.

EM and K-Means

In the previous lecture, we developed an EM algorithm for a Gaussian Mixture model - we can have another look at this algorithm and make some simplifying assumptions.

We can assume that the covariance for each mixture component is simply a fixed identity matrix.

There is therefore no need to worry about estimating these, if they are fixed.

We can also think about the posterior probabilities of the data points being allocated to a Gaussian component.

EM and K-Means

In this simplified case, where each Σ_k is a fixed identity matrix, then the E-step is simply

$$E\{z_{kn}\} = p(k|\mathbf{x}_n) \propto \exp\left(-\frac{1}{2}||\mathbf{x}_n - \mathbf{m}_k||^2\right)$$

and the M-step is just

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} p(k|\mathbf{x}_n)\mathbf{x}_n}{\sum_{m=1}^{N} p(k|\mathbf{x}_n)}$$

EM and K-Means

If we make a hard decision about the expected value of z_{kn} based on the maximum of the posterior, we should be able to see that the maximum of the posterior corresponds to the minimum of $||\mathbf{x}_n - \mathbf{m}_k||^2$. This is exactly what we are doing in K-means!

If we choose z_{kn} based on the maximum posterior, our M-step is exactly the cluster centre updates for K-means clustering.

K-means clustering can therefore be obtained directly from the EM algorithm using a mixture of unit variance isotropic Gaussians, where at the E-step a hard decision about component membership is made, rather than assigning a probability of membership.

Kernel K-Means

Kernel K-Means

Let's consider the clustering criterion that is used in K-means,

$$\begin{aligned}\mathcal{E}_K &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\mathbf{x}_n - \mathbf{m}_k||^2 \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n - \mathbf{m}_k)^T(\mathbf{x}_n - \mathbf{m}_k) \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}(\mathbf{x}_n^T\mathbf{x}_n - 2\mathbf{m}_k^T\mathbf{x}_n + \mathbf{m}_k^T\mathbf{m}_k)\end{aligned}$$

and in particular we can look at the terms $\mathbf{m}_k^T\mathbf{x}_n$ and $\mathbf{m}_k^T\mathbf{m}_k$.

Kernel K-Means

We can easily see that

$$\mathbf{m}_k^T\mathbf{x}_n = \frac{1}{N_k}\sum_{m=1}^{N} z_{km}\mathbf{x}_m^T\mathbf{x}_n$$

and

$$\begin{aligned}\mathbf{m}_k^T\mathbf{m}_k &= \left(\frac{1}{N_k}\sum_{p=1}^{N} z_{kp}\mathbf{x}_p\right)^2 \\ &= \frac{1}{N_k^2}\sum_{p=1}^{N}\sum_{l=1}^{N} z_{kp}z_{kl}\mathbf{x}_p^T\mathbf{x}_l\end{aligned}$$

We therefore see that the objective function can be defined purely in terms of *inner products* of pairs of data points.

Kernel K-Means

We can therefore define a more general kernel K-means algorithm that takes into account more flexible representations of the nonlinear relationships that might be present in the data.

$$\begin{aligned}\mathcal{E}_K^\phi &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}||\phi(\mathbf{x}_n) - \mathbf{m}_k^\phi||^2 \\ &= \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(K(\mathbf{x}_n, \mathbf{x}_n)\right) \\ &\quad - \sum_{k=1}^{K}\sum_{m=1}^{N} z_{km}\left(\frac{2}{N_k}\sum_{m=1}^{N} z_{km}K(\mathbf{x}_m, \mathbf{x}_n)\right) \\ &\quad - \sum_{k=1}^{K}\sum_{n=1}^{N} z_{kn}\left(\frac{1}{N_k^2}\sum_{p=1}^{N}\sum_{l=1}^{N} z_{kp}z_{kl}K(\mathbf{x}_p, \mathbf{x}_l)\right)\end{aligned}$$

We must also worry about the appropriate choice of parameters.

EM and K-Means

In the previous lecture, we developed an EM algorithm for a Gaussian Mixture model - we can have another look at this algorithm and make some simplifying assumptions.

We can assume that the covariance for each mixture component is simply a fixed identity matrix.

There is therefore no need to worry about estimating these, if they are fixed.

We can also think about the posterior probabilities of the data points being allocated to a Gaussian component.

EM and K-Means

In this simplified case, where each Σ_k is a fixed identity matrix, then the E-step is simply

$$E\{z_{kn}\} = p(k|\mathbf{x}_n) \propto \exp\left(-\frac{1}{2}||\mathbf{x}_n - \mathbf{m}_k||^2\right)$$

and the M-step is just

$$\mathbf{m}_k = \frac{\sum_{n=1}^{N} p(k|\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} p(k|\mathbf{x}_n)}$$

EM and K-Means

If we make a hard decision about the expected value of z_{kn} based on the maximum of the posterior, we should be able to see that the maximum of the posterior corresponds to the minimum of $||\mathbf{x}_n - \mathbf{m}_k||^2$. This is exactly what we are doing in K-means!

If we choose z_{kn} based on the maximum posterior, our M-step is exactly the cluster centre updates for K-means clustering.

K-means clustering can therefore be obtained directly from the EM algorithm using a mixture of unit variance isotropic Gaussians, where at the E-step a hard decision about component membership is made, rather than assigning a probability of membership.

Conclusions

K-means performs *hard clustering* - each data point is assigned to only 1 cluster.

A mixture model can be thought of as performing *soft clustering* - it returns a posterior probability for each cluster.

Although K-means is a geometric clustering algorithm developed using intuition about distances, we can interpret this as a special case of a mixture of Gaussian distributions with hard assignment of data points to components.

M5MS10
Machine Learning

Spring 2018

Lecture 9

Dr Ben Calderhead

b.calderhead@imperial.ac.uk