

# Lecture 13. Clustering. Gaussian Mixture Model

COMP90051 Statistical Machine Learning

Semester 2, 2017  
Lecturer: Andrey Kan



THE UNIVERSITY OF  
MELBOURNE

# This lecture

- Unsupervised learning
  - \* Diversity of problems
  - \* Pipelines
- Clustering
  - \* Problem formulation
  - \* Algorithms
  - \* Choosing the number of clusters
- Gaussian mixture model (GMM)
  - \* A probabilistic approach to clustering
  - \* GMM clustering as an optimisation problem

# Unsupervised Learning

A large branch of ML that concerns  
with learning the structure of the  
data in the absence of labels

# Previously: Supervised learning

- Supervised learning: Overarching aim is making predictions from data
- We studied methods such as random forest, ANN and SVM in the context of this aim
- We had instances  $\mathbf{x}_i \in \mathbf{R}^m$ ,  $i = 1, \dots, n$  and corresponding labels  $y_i$  as inputs, and the aim was to predict labels for new instances
- Can be viewed as a function approximation problem, but with a big caveat
- The ability to generalise is critical

# Now: Unsupervised learning

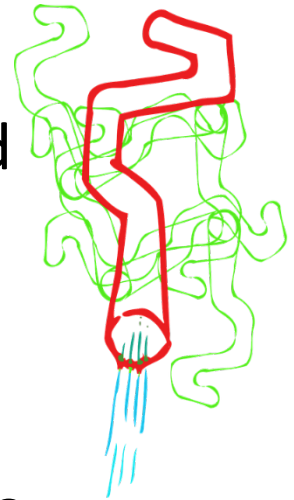
- Next few lectures: unsupervised learning methods
- In unsupervised learning, there is no dedicated variable called a “label”
- Instead, we just have a set of points  $\mathbf{x}_i \in \mathbf{R}^m$ ,  $i = 1, \dots, n$ 
  - \* And often data cannot be reduced to points in  $\mathbf{R}^m$  (e.g., data is a set of variable length sequences)
- The aim of unsupervised learning is to explore the structure (patterns, regularities) of the data
- The aim of “exploring the structure” is vague

# Applications of unsupervised learning

- Diversity of tasks fall into unsupervised learning category
- This subject covers some common applications of unsupervised learning:
  - \* Clustering (this week)
  - \* Dimensionality reduction (next week)
  - \* Learning parameters of probabilistic models (after break)
- A few other applications not covered in this course:
  - \* Market basket analysis. E.g., use supermarket transaction logs to find items that are frequently purchased together
  - \* Outlier detection. E.g., find potentially fraudulent credit card transactions

# Data analysis pipelines

- Clustering and dimensionality reduction are listed as separate problems
- In many cases, these can be viewed as distinct unsupervised learning tasks
- However, different methods can be combined into powerful data analysis *pipelines*
- E.g., we will use these pipelines for data points where patterns are obscure, or for datasets that are not points in  $\mathbf{R}^m$
- Before getting to this stage, we first consider a classical clustering problem



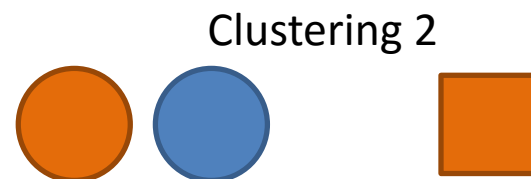
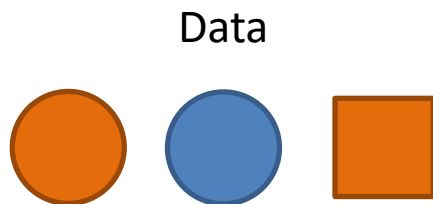
# Clustering

A fundamental task in Machine Learning. Thousands of algorithms, yet no definitive universal solution



# Introduction to clustering

- Clustering is automatic grouping of objects such that the objects within each group (cluster) are more similar to each other than objects from different groups
  - \* An extremely vague definition that reflects the variety of real-world problems that require clustering
- The key to this definition is defining what “*similar*” means



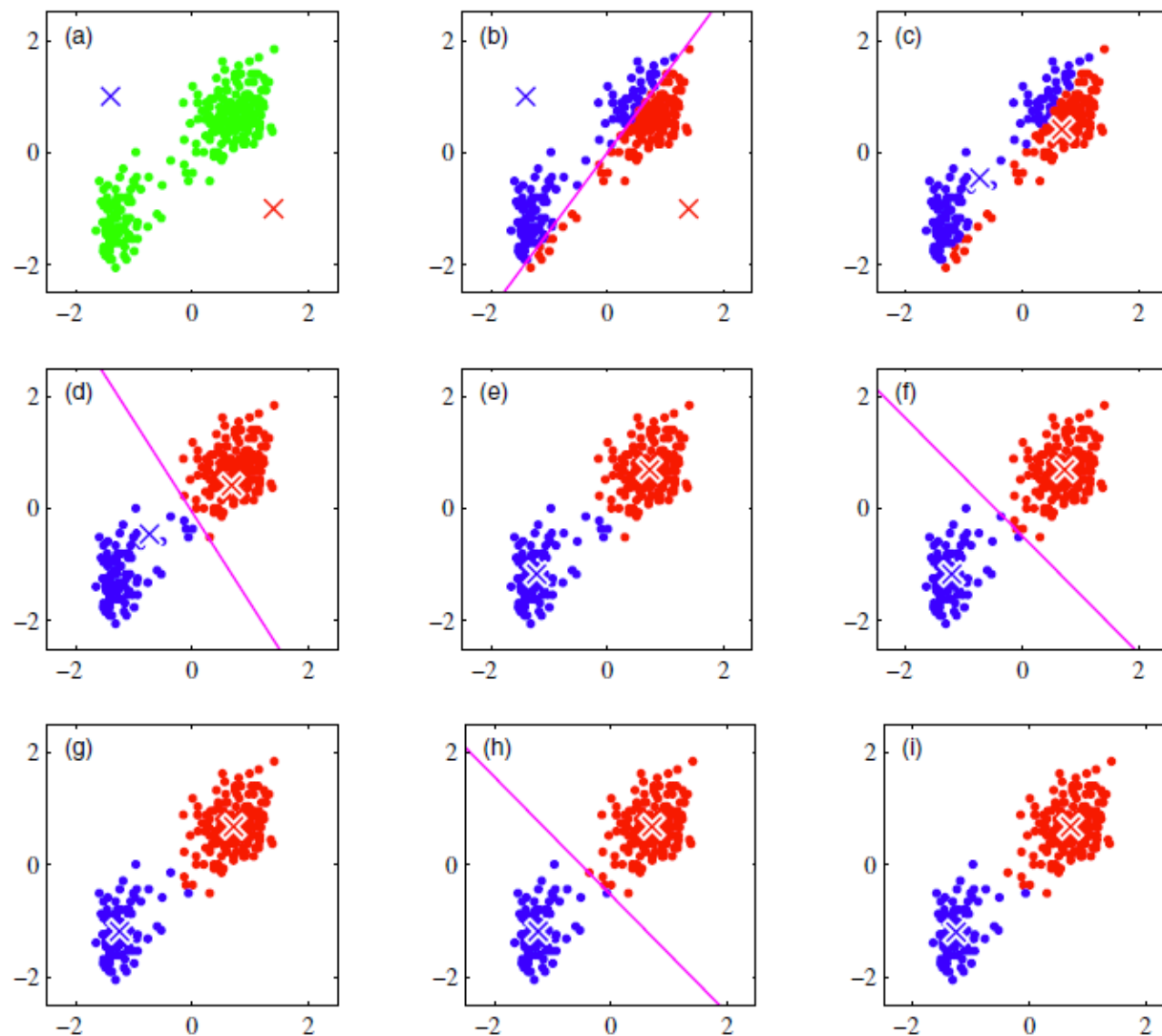
# Measuring (dis)similarity

- Consider a “classical” setting where the data is a set of points  $\mathbf{x}_i \in \mathbf{R}^m$ ,  $i = 1, \dots, n$
- Instead of “similarity”, it is sometimes more convenient to use an opposite concept of “dissimilarity” or “difference”
- A natural choice for formalising the “difference” between a pair of points is Euclidean distance

$$d_{ij} \equiv \|\mathbf{x}_i - \mathbf{x}_j\| = \sqrt{\sum_{l=1}^m \left( (\mathbf{x}_i)_l - (\mathbf{x}_j)_l \right)^2}$$

- Here  $\mathbf{x}_i$  is a vector and  $(\mathbf{x}_i)_l$  denotes the  $l$ -th element of that vector

# Refresher on K-means clustering



Requires specifying the number of clusters in advance

Measures “dissimilarity” using Euclidean distance

Finds “spherical” clusters

An iterative optimization procedure

Data: Old Faithful  
Geyser Data: waiting time between eruptions and the duration of eruptions

# K-means as iterative optimisation

1. Initialisation: choose  $k$  cluster centroids randomly
2. Update:
  - a) Assign points to the nearest centroid
  - b) Compute centroids under the current assignment
3. Termination: if no change then **stop**
4. Go to **Step 2**

# Clustering algorithms

- There are thousands (!) of clustering algorithms
  - \* *Jain A.K. et al., Data clustering: 50 years beyond K-means, 2010, Pattern recognition letters*
- K-means is still one of the most popular algorithms (perhaps the most popular)
  - \* *Wu X. et al., Top 10 algorithms in data mining, 2008, Knowledge and information systems*
- Many popular algorithms do not require specifying  $k$ 
  - \* Hierarchical clustering (*e.g., see Hastie et al., The elements of statistical learning*)
  - \* DBSCAN (*Ester M. et al., A density-based algorithm for discovering clusters in large spatial databases with noise, 1996, KDD conference*)
  - \* Affinity propagation (*Brendan J.F. and Dueck D., Clustering by passing messages between data points, 2007, Science*)

# Consensus clustering (1/3)

- Unsupervised learning counterpart of bagging
  - \* *Monti S. et al., Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data, 2003, Machine Learning*
- Consider a dataset  $D = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ , and some pre-defined number of iterations  $T$
- The algorithm creates sampled versions  $D_1, \dots, D_T$  of the original data
  - \* In principle, one can use bootstrapping (sampling with replacement), resulting in  $|D_t| = n$
  - \* For consensus clustering authors consider subsampling (without replacement), resulting in  $|D_t| < n$

# Consensus clustering (2/3)

- $D_1, \dots, D_T$  are sampled versions of original data
- Define an  $n \times n$  indicator matrix  $\mathbf{I}_t$ , such that  $\mathbf{I}_t(i, j) = 1$  if  $\mathbf{x}_i, \mathbf{x}_j \in D_t$  and  $\mathbf{I}_t(i, j) = 0$  otherwise
- Apply clustering independently on each  $D_t$
- Define an  $n \times n$  association matrix  $\mathbf{M}_t$ , such that  $\mathbf{M}_t(i, j) = 1$  if  $\mathbf{x}_i, \mathbf{x}_j$  are in the same cluster, and  $\mathbf{M}_t(i, j) = 0$  otherwise
  - \* If  $\mathbf{x}_i \notin D_t$  or  $\mathbf{x}_j \notin D_t$  then  $\mathbf{M}_t(i, j) = 0$
- The consensus matrix  $\mathcal{M}$  is defined as  $\mathcal{M}(i, j) \equiv \frac{\sum_{t=1}^T \mathbf{M}_t(i, j)}{\sum_{t=1}^T \mathbf{I}_t(i, j)}$ 
  - \* Proportion of clustering runs in which two items cluster together

# Consensus clustering (3/3)

- Algorithm

1. Choose a clustering algorithm
2. For  $t = 1 \dots T$ 
  - a) Sample  $D_t$  from  $D$  (this can also be done by sampling features rather than data points)
  - b) Apply clustering on  $D_t$
  - c) Save connectivity matrix  $\mathbf{M}_t$  and indicator matrix  $\mathbf{I}_t$
3. Construct consensus matrix  $\mathcal{M}$
4. Cluster  $D$  using  $\mathcal{M}$  as a similarity matrix (e.g., apply hierarchical clustering)

Compare with  
random forest  
generation



# Choosing the number of clusters

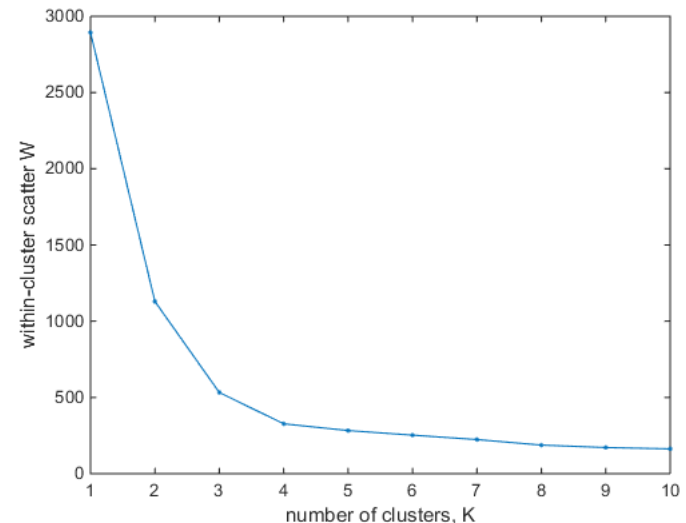
- Many clustering algorithms do not require  $k$ , but require specifying some other parameters that influence resulting number of clusters
- Suppose that we are using the algorithm that does require  $k$
- The number of clusters can be known from context
  - \* E.g., clustering genetic profiles from a group of cells that is known to contain a certain number of cell types
- Visualising the data (e.g., using multidimensional reduction, next week) can help to estimate the number of clusters
- Another strategy is to try a few plausible values and see whether it makes difference to the analysis

# Number of clusters and overfitting

- In the context of k-means it might be appealing to treat  $k$  as an parameter to be optimised
- This does not work because larger  $k$  results in a more flexible model that will always fit the data better
  - \* This is analogous to overfitting in supervised learning
- Instead, approaches that can work are:
  - \* Cross-validation-like strategies for determining  $k$
  - \* Try several possible  $k$  and look at the trend
  - \* Information-theoretic results
- Example of the second approach is shown in the next slide
- Examples of the third approach are given in green slides after that

# Kink method and gap statistics

- Manual inspection of minimised within cluster variation as a function of  $k$
- The real number of clusters is indicated by the kink in this curve
- The gap statistics (*Hastie et al. book*) developed for k-means clustering is an automated version of the kink method
- This method measures the gaps between each  $k$  and analyses their distribution



# Akaike information criterion

- This and next methods are only applicable for parametric probabilistic models  $p(\mathbf{X}|\boldsymbol{\theta})$ . Let  $\hat{\boldsymbol{\theta}}$  be MLE for this model, and let  $L^* \equiv p(\mathbf{X}|\hat{\boldsymbol{\theta}})$
- Akaike information criterion (AIC) is defined as
$$AIC = 2N_{par} - 2 \ln L^*$$
- Here  $N_{par}$  is the number of free parameters
- The equation is simple, but the derivation is very complicated. AIC is one of fundamental results in statistic
- AIC estimates divergence between the true unknown model (e.g., GMM with true number of clusters), and the current model

# Akaike information criterion

- Method: consider several different  $k$ , and their corresponding GMM. Find MLE parameters for each model
- Compute AIC for each model and choose  $k$  that resulted in the smallest AIC
- The smallest AIC is preferable because AIC is an estimator of divergence between the true and current models
- AIC estimator was shown to be biased for finite sample sizes, thus a correction has been proposed which should be used in practice instead AIC ( $n$  is the number of data points)

$$AICc = 2N_{par} + \ln L^* + \frac{2N_{par}(N_{par} + 1)}{n - N_{par} - 1}$$

# Bayesian information criterion

- AIC was derived from a frequentist standpoint. Bayesian information criterion (BIC) represents the Bayesian approach to model selection
- Bayesian model selection is based on marginal likelihood  $p(data|model)$
- BIC is an approximate computation of the marginal likelihood
- BIC is defined as
$$BIC = N_{par} \ln n - 2 \ln L^*$$
- One should choose a model with the smallest BIC

# Gaussian Mixture Model

A probabilistic view of clustering

# Why GMM clustering

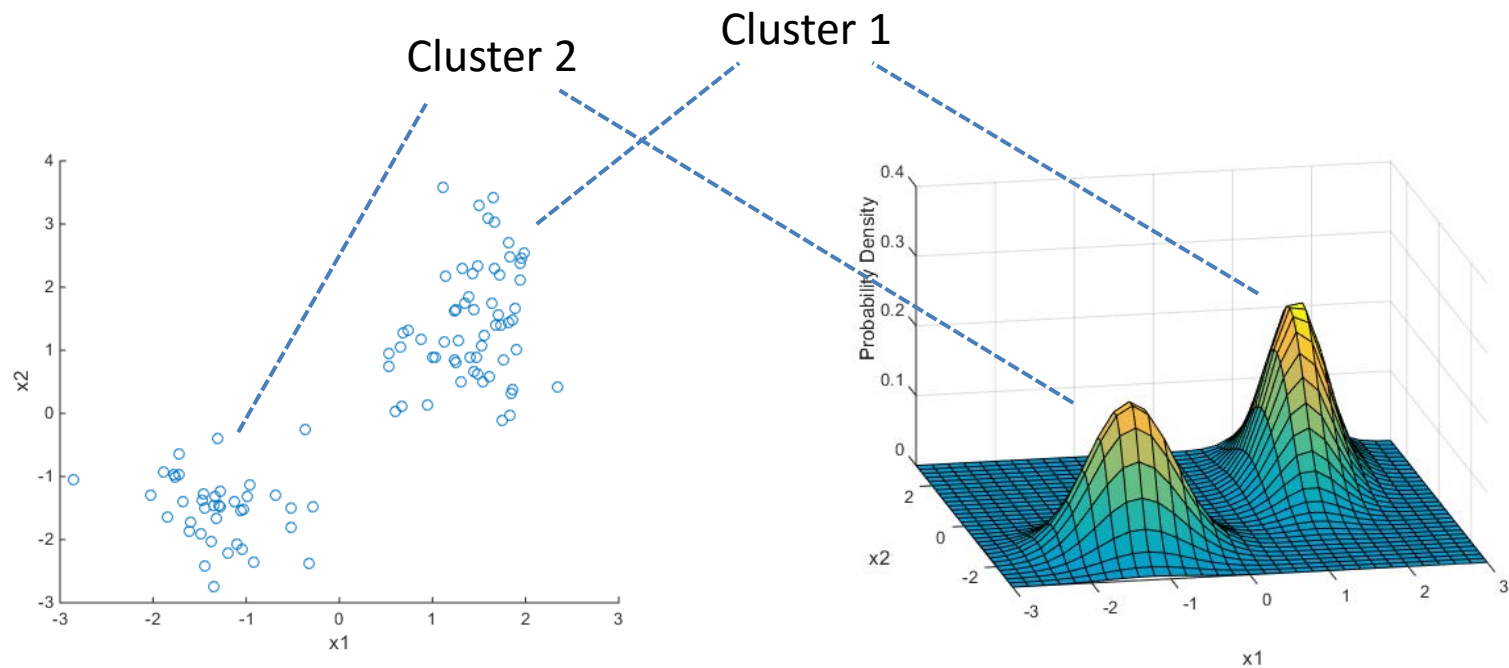
- K-means algorithm is one of the most popular algorithms, GMM clustering is a generalisation of k-means
- Empirically, works well in many cases
  - \* Moreover, it can be used in a manifold learning pipeline (coming soon)
- Reasonably simple and mathematically tractable
- Example of a probabilistic approach
- Example application of Expectation Maximisation (EM) algorithm
  - \* EM algorithm is a generic technique, not only for GMM clustering



# Clustering: Probabilistic interpretation

Clustering can be viewed as identification of components of a probability density function that generated the data

Identifying cluster centroids can be viewed as finding modes of distributions



# Modelling uncertainty in data clustering

- K-means clustering assigns each point to exactly one cluster
  - \* In other words, the result of such a clustering is partitioning into  $k$  subsets
- Similar to k-means, a probabilistic mixture model requires the user to choose the number of clusters in advance
- Unlike k-means, the probabilistic model gives us a power to express uncertainty about the origin of each point
  - \* Each point originates from cluster  $c$  with probability  $w_c$ ,  $c = 1, \dots, k$
- That is, each point still originates from one particular cluster (aka component), but we are not sure from which one
- Next, for each individual component, the normal distribution is a common modelling choice

# Normal (aka Gaussian) distribution

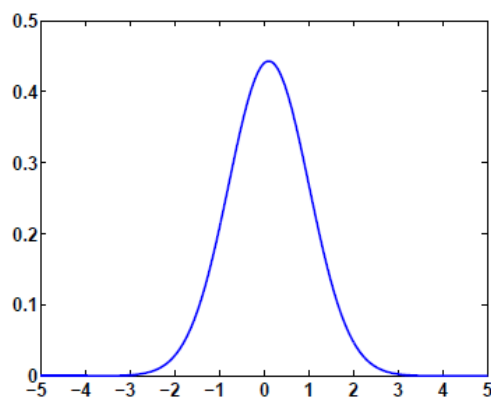
- Recall that a 1D Gaussian is

$$\mathcal{N}(x|\mu, \sigma) \equiv \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

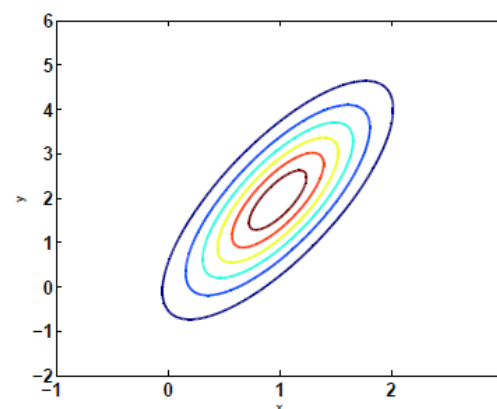
- And a  $m$ -dimensional Gaussian is

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \equiv (2\pi)^{-\frac{m}{2}} (\det \boldsymbol{\Sigma})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right)$$

- \*  $\boldsymbol{\Sigma}$  is a symmetric  $m \times m$  matrix that is assumed to be positive definite
- \*  $\det \boldsymbol{\Sigma}$  denotes matrix determinant



(a) 1-Dim



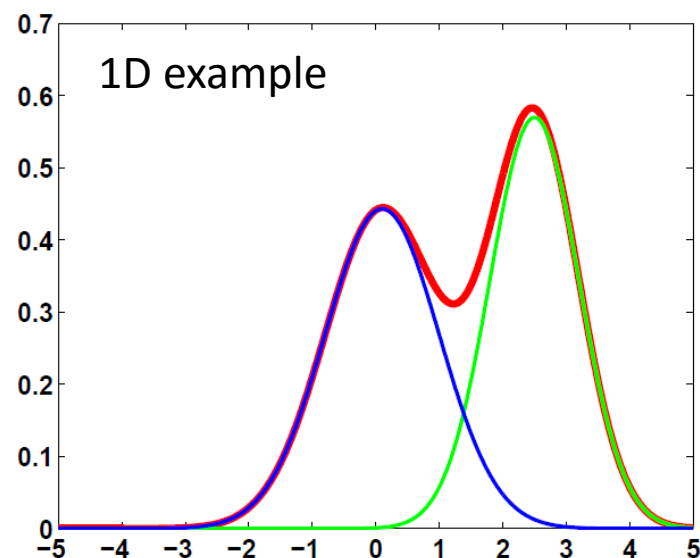
(b) 2-Dim

# Gaussian mixture model (GMM)

- Gaussian mixture distribution (for one data point):

$$p(\mathbf{x}) \equiv \sum_{c=1}^k w_c \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

- Here  $w_c \geq 0$  and  $\sum_{c=1}^k w_c = 1$
- That is,  $w_1, \dots, w_k$  is a probability distribution over components
- Parameters of the model are  $w_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \dots, k$



Mixture and individual component densities  
are re-scaled for visualisation purposes

Figure: Bishop

# Checkpoint

- Consider a GMM with five components for 3D data. How many independent parameters does this model have?



$$6 \times 5 + 3 \times 5 + 4$$



$$6 \times 5 + 3 \times 5 + 5$$



$$9 \times 5 + 3 \times 5 + 5$$

art: OpenClipartVectors at  
pixabay.com (CC0)



# Clustering as an optimisation problem

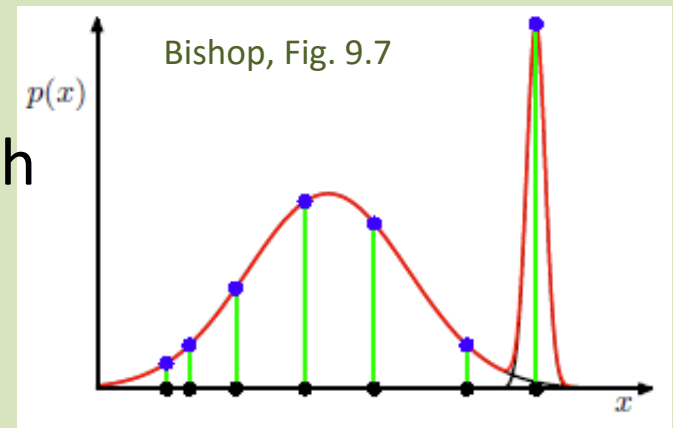
- Given a set of data points, we assume that data points are generated by a GMM
  - \* Each point in our dataset originates from the  $c$ -th normal distribution component with probability  $w_c$
- Clustering now amounts to finding parameters of the GMM that “best explain” the observed data
- But what does “best explain” mean?
- We are going to call upon another old friend: MLE principle tells us to use parameter values that maximise  $p(\mathbf{x}_1, \dots, \mathbf{x}_n)$

# Fitting a GMM model to data

- Assuming that data points are independent, our aim is to find  $w_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \dots, k$  that maximise

$$p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \prod_{i=1}^n \sum_{c=1}^k w_c \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

- This is actually an ill-posed problem
  - \* Singularities (points at which the likelihood is not defined)
  - \* Non-uniqueness
- Theoretical cure – Bayesian approach
- Practical cure – heuristically avoid singularities



# Fitting a GMM model to data

- Yet again, we are facing an optimisation problem
- Assuming that data points are independent, our aim is to find  $w_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \dots, k$  that maximise

$$p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \prod_{i=1}^n \sum_{c=1}^k w_c \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

- Let's see if this can be solved analytically
- Taking the derivative of this expression is pretty awkward, try the usual log trick



# Attempting the log trick for GMM

- Our aim is to find  $w_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \dots, k$  that maximise

$$\log p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \sum_{i=1}^n \log \left( \sum_{c=1}^k w_c \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \right)$$

- The log cannot be pushed inside the sum. The derivative of the log likelihood still going to have a cumbersome form
- We should use an iterative procedure

# Fitting a GMM using iterative optimisation

- So there's little prospect in analytical solution
- At this point, we could use the gradient descent algorithm
- But it still requires taking partial derivatives
- Another problem of using the gradient descent are complicated constraints on parameters
- We aim to find  $w_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \dots, k$ , where  $\boldsymbol{\Sigma}_c$  are symmetric and positive definite for each  $c$ , and where  $w_c$  add up to one

# Introduction to Expectation Maximisation

- Expectation Maximisation (EM) algorithm is a common way to find parameters of a GMM
- EM is a generic algorithm for finding MLE of parameters of a probabilistic model
- Broadly speaking, as “input” EM requires
  - \* A probabilistic model that is can be specified by a fixed number of parameters
  - \* Data 😊
- EM is widely used outside clustering and GMMs (another application of EM coming soon)

# MLE vs EM

- MLE is a frequentist *principle* that suggests that given a dataset, the “best” parameters to use are the ones that maximise the probability of the data
  - \* MLE is a way *to formally pose* the problem
- EM is an *algorithm*
  - \* EM is a way *to solve* the problem posed by MLE
- MLE can be found by other methods such as gradient descent (but gradient descent is not always the most convenient method)

# This lecture

- Unsupervised learning
  - \* Diversity of problems
  - \* Pipelines
- Clustering
  - \* Problem formulation
  - \* Algorithms
  - \* Choosing the number of clusters
- Gaussian mixture model (GMM)
  - \* A probabilistic approach to clustering
  - \* GMM clustering as an optimisation problem