Outline

- Flat clustering
  - Mixture of Gaussians
  - K-means
- Hierarchical clustering
  - bottom-up
- Spectral based clustering
- Applications
Clustering

- Given set of data points, group them
- Unsupervised learning
- Learn the similarity. Which patient are similar? (or customers, faces, earthquakes, …)
Clustering vs. Classification

- **Clustering**
  - **Instance:** $\{x_i\}_{i=1}^N$
  - **Learn:** $<x_i, t_i>$ and/or mapping from $x$ to $t(x)$

- **Classification/Regression**
  - **Instance:** $<x_i, t_i>$
  - **Learn:** mapping from $x$ to $t(x)$
Clustering: image segmentation

Mean-shift segmentation
Mixtures of Gaussians

- Mixture distribution:
  - Assume $P(x)$ is a mixture of $K$ different Gaussians
  - Assume each data point, $x$, is generated by 2-step process
    - Choose one of the $K$ Gaussians as label $z$
    - Generate $x$ according to the Gaussian $N(\mu_z, \Sigma_z)$

$$P(x) = \sum_{z=1}^{K} P(Z = z \mid \pi) N(x \mid \mu_z, \Sigma_z)$$

- What object function shall we optimize?
  - Maximize data likelihood
Mixtures of Gaussians (cont.)

- Multivariate Gaussian model

\[ p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\} \]

- How to generate it?

\[ F_{\mu, \sigma^2}(x) = \int_{-\infty}^{x} p(z|\mu, \sigma^2) dz \]

\[ u \sim \text{Uniform}(0, 1) \Rightarrow x = F_{\mu, \sigma^2}^{-1}(u) \sim p(x|\mu, \sigma^2) \]

\[ z_i \sim p(z_i|\mu = 0, \sigma^2 = 1), \; z = [z_1, \ldots, z_d]^T \]

\[ x = \Sigma^{1/2} z + \mu \]
Multi-variate density estimation

- A mixture of Gaussians model

\[ p(x|\theta) = \sum_{i=1}^{k} p_j p(x|\mu_j, \Sigma_j) \]

where \( \theta = \{p_1, \ldots, p_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k\} \) contains all the parameters of the mixture model. \( \{p_j\} \) are known as mixing proportions or coefficients.
Mixtures of Gaussians: Wishart distribution

- A mixture of Gaussian Model:

\[
p(x|\theta) = \sum_{i=1}^{k} p_j p(x|\mu_j, \Sigma_j)
\]

\[
\theta = \{p_1, \ldots, p_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k\}
\]

- Wishart prior

\[
P(\Sigma|S, n') \propto \frac{1}{\det(\Sigma)^{n'/2}} \exp\left(-\frac{n'}{2}\text{Trace}(\Sigma^{-1} S)\right)
\]

\(S\) = “prior” covariance matrix

\(n'\) = equivalent sample size
Mixture density

- Data generation process:

\[
p(x|\theta) = \sum_{j=1,2} P(y = j) \cdot p(x|y = j) \quad \text{(generic mixture)}
\]

\[
= \sum_{j=1,2} p_j \cdot p(x|\mu_j, \Sigma_j) \quad \text{(mixture of Gaussians)}
\]

- Any data point \( x \) could have been generated in two ways
Mixture density

- If we are given just \( x \) we don’t know which mixture component this example came from

\[
p(x|\theta) = \sum_{j=1,2} p_j p(x|\mu_j, \Sigma_j)
\]

- We can evaluate the posterior probability that an observed \( x \) was generated from the first mixture component

\[
P(y = 1|x, \theta) = \frac{P(y = 1) \cdot p(x|y = 1)}{\sum_{j=1,2} P(y = j) \cdot p(x|y = j)}
= \frac{p_1 p(x|\mu_1, \Sigma_1)}{\sum_{j=1,2} p_j p(x|\mu_j, \Sigma_j)}
\]

- This solves a credit assignment problem
Consider sampling $x$ from the mixture density, then $y$ from the posterior over the components given $x$, and finally $x'$ from the component density indicated by $y$:

\[
\begin{align*}
x & \sim p(x|\theta) \\
y & \sim P(y|x, \theta) \\
x' & \sim p(x'|y, \theta)
\end{align*}
\]

Is $y$ a fair sample from the prior distribution $P(y)$?
Is $x'$ a fair sample from the mixture density $p(x'|\theta)$?
Mixture density estimation

- Suppose we want to estimate a two component mixture of Gaussians model.

\[ p(x|\theta) = p_1 p(x|\mu_1, \Sigma_1) + p_2 p(x|\mu_2, \Sigma_2) \]

- If each example \( x_i \) in the training set were labeled \( y_i = 1, 2 \) according to which mixture component (1 or 2) had generated it, then the estimation would be easy.

- Labeled examples \( \Rightarrow \) no credit assignment problem
Mixture density estimation

When examples are already assigned to mixture components (labeled), we can estimate each Gaussian independently.

- If $\hat{n}_j$ is the number of examples labeled $j$, then for each $j = 1, 2$ we set

$$\hat{p}_j \leftarrow \frac{\hat{n}_j}{n}$$

$$\hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} x_i$$

$$\hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T$$
Mixture density estimation:
credit assignment

- Of course we don’t have such labels … but we can guess what the labels might be based on our current mixture distribution.

- We get soft labels or posterior probabilities of which Gaussian generated which example:

  \[ \hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta) \]

  where \[ \sum_{j=1,2} \hat{p}(j|i) = 1 \] for all \[ i = 1, \ldots, n. \]

- When the Gaussians are almost identical (as in the figure), \[ \hat{p}(1|i) \approx \hat{p}(2|i) \] for almost any available point \( x_i \).

  Even slight differences can help us determine how we should modify the Gaussians.
The EM algorithm

**E-step:** softly assign examples to mixture components
\[ \hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta), \quad \text{for all } j = 1, 2 \text{ and } i = 1, \ldots, n \]

**M-step:** re-estimate the parameters (separately for the two Gaussians) based on the soft assignments.

\[ \hat{n}_j \leftarrow \sum_{i=1}^{n} \hat{p}(j|i) = \text{Soft \# of examples labeled } j \]

\[ \hat{p}_j \leftarrow \frac{\hat{n}_j}{n} \]

\[ \hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) x_i \]

\[ \hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T \]
The EM-algorithm

- Each iteration of the EM-algorithm \textit{monotonically} increases the (log-)likelihood of the $n$ training examples $x_1, \ldots, x_n$:

$$
\log p(\text{data} | \theta) = \sum_{i=1}^{n} \log \left( \frac{p(x_i | \theta)}{p_1 p(x_i | \mu_1, \Sigma_1) + p_2 p(x_i | \mu_2, \Sigma_2)} \right)
$$

where $\theta = \{p_1, p_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$ contains all the parameters of the mixture model.
The EM algorithm

- The EM-algorithm finds a local maximum of $l(\theta; D)$

**E-step:** evaluate the expected complete log-likelihood

$$J(\theta; \theta^{(t)}) = \sum_{i=1}^{n} E_{j \sim P(j|x_i, \theta^{(t)})} \log \left( p_j p(x_i | \mu_j, \Sigma_j) \right)$$

$$= \sum_{i=1}^{n} \sum_{j=1,2} P(j| x_i, \theta^{(t)}) \log \left( p_j p(x_i | \mu_j, \Sigma_j) \right)$$

**M-step:** find the new parameters by maximizing the expected complete log-likelihood

$$\theta^{(t+1)} \leftarrow \arg\max_{\theta} J(\theta; \theta^{(t)})$$
Regularized EM algorithm

- To maximize a penalized (regularized) log-likelihood

\[ l'(\theta; D) = \sum_{i=1}^{n} \log p(x_i|\theta) + \log p(\theta) \]

we only need to modify the M-step of the EM-algorithm. Specifically, in the M-step, we find \( \theta \) that maximize a penalized expected complete log-likelihood:

\[ J(\theta; \theta^{(t)}) = \sum_{i=1}^{n} E_{j \sim P(j|x_i, \theta^{(t)})} \log \left( p_j p(x_i|\mu_j, \Sigma_j) \right) \]
\[ + \log p(p_1, p_2) + \log p(\Sigma_1) + \log p(\Sigma_1) \]

where, for example, \( p(p_1, p_2) \) could be a Dirichlet and each \( p(\Sigma_j) \) a Wishart prior.
Selecting the number of components

- As a simple strategy for selecting the appropriate number of mixture components, we can find $k$ that minimize the following asymptotic approximation to the description length:

$$DL \approx -\log p(\text{data}|\hat{\theta}_k) + \frac{d_k}{2} \log(n)$$

where $n$ is the number of training points, $\hat{\theta}_k$ is the maximum likelihood parameter estimate for the $k$-component mixture, and $d_k$ is the (effective) number of parameters in the $k$-mixture.
Mixture density estimation: example
K-means clustering

Given data \( \langle x_1 \ldots x_n \rangle \), and K, assign each \( x_i \) to one of K clusters, \( C_1 \ldots C_K \), minimizing

\[
J = \sum_{j=1}^{K} \sum_{x_i \in C_j} ||x_i - \mu_j||^2
\]

Where \( \mu_j \) is mean over all points in cluster \( C_j \).

K-Means Algorithm:

1. Initialize \( \mu_1 \ldots \mu_K \) randomly
2. Repeat until convergence:
   1. Assign each point \( x_i \) to the cluster with the closest mean \( \mu_j \)
   2. Calculate the new mean for each cluster
      \[
      \mu_j \leftarrow \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i
      \]
K-Means vs. Mixture of Gaussians

- Both are iterative algorithms to assign points to clusters

- Objective function
  - K Means: minimize
  - MoG: maximize likelihood

- MoG the more general formulation
  - Equivalent to K Means when $\Sigma_k = \sigma I$, and $\sigma \to 0$
Hierarchical (bottom-up) clustering

- Hierarchical agglomerative clustering: we sequentially merge the pair of “closest” points/clusters

- The procedure
  1. Find two closest points (clusters) and merge them
  2. Proceed until we have a single cluster (all the points)

- Two prerequisites:
  1. distance measure $d(x_i, x_j)$ between two points
  2. distance measure between clusters (cluster linkage)
Hierarchical (bottom-up) clustering

- A *linkage* method: we have to be able to measure distances between clusters of examples $C_k$ and $C_l$
  
a) Single linkage:

$$d_{kl} = \min_{i \in C_k, j \in C_l} d(x_i, x_j)$$

b) Average linkage:

$$d_{kl} = \frac{1}{|C_l||C_k|} \sum_{i \in C_k, j \in C_l} d(x_i, x_j)$$

c) Centroid linkage:

$$d_{kl} = d(\bar{x}_k, \bar{x}_l), \quad \bar{x}_l = \frac{1}{|C_l|} \sum_{i \in C_l} x_i$$
Hierarchical (bottom-up) clustering

- A dendrogram representation of hierarchical clustering

The height of each pair represents the distance between the merged clusters; the specific linear ordering of points is chosen for clarity.
Spectral clustering

- The spectral clustering method we define relies on a random walk representation over the points. We construct this in three steps:
  1. a nearest neighbor graph
  2. similarity weights on the edges:
     \[ W_{ij} = \exp\{-\beta \|x_i - x_j\|\} \]
     where \( W_{ii} = 1 \) and the weight is zero for non-edges.
  3. transition probability matrix
     \[ P_{ij} = \frac{W_{ij}}{\sum_{j'} W_{ij'}} \]
Properties of the random walk

- If we start from \( i_0 \), the distribution of points \( i_t \) that we end up in after \( t \) steps is given by

\[
\begin{align*}
  i_1 & \sim \pi_{i_0 i_1}, \\
  i_2 & \sim \sum_{i_1} \pi_{i_0 i_1} \pi_{i_1 i_2} = [P^2]_{i_0 i_2}, \\
  i_3 & \sim \sum_{i_1} \sum_{i_2} \pi_{i_0 i_1} \pi_{i_1 i_2} \pi_{i_2 i_3} = [P^3]_{i_0 i_3}, \\
  & \vdots \\
  i_t & \sim [P^t]_{i_0 i_t}
\end{align*}
\]

where \( P^t = PP \ldots P \) (\( t \) matrix products) and \([\cdot]_{ij}\) denotes the \( i, j \) component of the matrix.
Random walk and clustering

- The distributions of points we end up in after \( t \) steps converge as \( t \) increases. If the graph is connected, the resulting distribution is independent of the starting point.

Even for large \( t \), the transition probabilities \( [P^t]_{ij} \) have a slightly higher probability of transitioning within “clusters” than across; we want to recover this effect from eigenvalues/vectors.
Eigenvalues/vectors and spectral clustering

- Let $W$ be the matrix with components $W_{ij}$ and $D$ a diagonal matrix such that $D_{ii} = \sum_j W_{ij}$. Then
  \[ P = D^{-1}W \]

- To find out how $P^t$ behaves for large $t$ it is useful to examine the eigen-decomposition of the following symmetric matrix
  \[ D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = \lambda_1 z_1 z_1^T + \lambda_2 z_2 z_2^T + \ldots + \lambda_n z_n z_n^T \]
  where the ordering is such that $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$. 
Eigenvalues/vectors cont’d

- The symmetric matrix is related to $P^t$ since

$$
(D^{-rac{1}{2}}WD^{-rac{1}{2}}) \cdots (D^{-rac{1}{2}}WD^{-rac{1}{2}}) = D^{rac{1}{2}} (P \cdots P) D^{-rac{1}{2}}
$$

This allows us to write the $t$ step transition probability matrix in terms of the eigenvalues/vectors of the symmetric matrix

$$
P^t = D^{-rac{1}{2}} \left( D^{-rac{1}{2}}WD^{-rac{1}{2}} \right)^t D^{rac{1}{2}}
$$

$$
= D^{-rac{1}{2}} \left( \lambda_1^t z_1 z_1^T + \lambda_2^t z_2 z_2^T + \ldots + \lambda_n^t z_n z_n^T \right) D^{rac{1}{2}}
$$

where $\lambda_1 = 1$ and

$$
P^\infty = D^{-rac{1}{2}} \left( z_1 z_1^T \right) D^{rac{1}{2}}$$
Eigenvalues/vectors and spectral clustering

- We are interested in the largest correction to the asymptotic limit

\[ P^t \approx P^\infty + D^{-\frac{1}{2}} \left( \lambda_2^t \mathbf{z}_2 \mathbf{z}_2^T \right) D^{\frac{1}{2}} \]

Note: \( [\mathbf{z}_2 \mathbf{z}_2^T]_{ij} = \mathbb{z}_{2i} \mathbb{z}_{2j} \) and thus the largest correction term increases the probability of transitions between points that share the same sign of \( \mathbb{z}_{2i} \) and decreases transitions across points with different signs.

- Binary spectral clustering: we divide the points into clusters based on the sign of the elements of \( \mathbf{z}_2 \)

\[ \mathbb{z}_{2j} > 0 \Rightarrow \text{cluster 1}, \text{otherwise cluster 0} \]
Spectral clustering: example

Components of the eigenvector corresponding to the second largest eigenvalue
Reference papers of SC


- And more about image segmentations …
  - Graph cut
  - Mean-shift
An example: ISO/BLE-charts

- **ISO-Charts:**
  - ISOMAP + Spectral Clustering + Stretch Minimization

- **BLE-Charts:**
  - Statistical Embedding + Spectral Clustering + Stretch Minimization