Unsupervised Learning

Andrea Passerini
passerini@disi.unitn.it

Machine Learning

Unsupervised Learning

Setting

- Supervised learning requires the availability of labelled examples
- Labelling examples can be an extremely expensive process
- Sometimes we don’t even know how to label examples
- *Unsupervised* techniques can be employed to group examples into **clusters**

k-means clustering

Setting

- Assumes examples should be grouped into *k* clusters
- Each cluster *i* is represented by its mean $\boldsymbol{\mu}_i$

Algorithm

1. Initialize cluster means $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$
2. Iterate until no mean changes:
    1. Assign each example to cluster with nearest mean
    2. Update cluster means according to assigned examples

How can we define (dis)similarity between examples ?

(Dis)similarity measures

- Standard Euclidean distance in $\mathbb{R}^d$:

$$d(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{\sum_{i=1}^{d} (x_i - x_i')^2}$$

- Generic Minkowski metric for $p \geq 1$:

$$d(\boldsymbol{x}, \boldsymbol{x}') = \left(\sum_{i=1}^{d} |x_i - x_i'|^p \right)^{1/p}$$

- Cosine similarity (cosine of the angle between vectors):

$$s(\boldsymbol{x}, \boldsymbol{x}') = \frac{\boldsymbol{x}^T \boldsymbol{x}'}{||\boldsymbol{x}||||\boldsymbol{x}'||}$$

How can we define quality of obtained clusters ?

Sum-of-squared error criterion

- Let n_i be the number of samples in cluster $\mathcal{D}_i$
- Let $\boldsymbol{\mu}_i$ be the cluster sample mean:

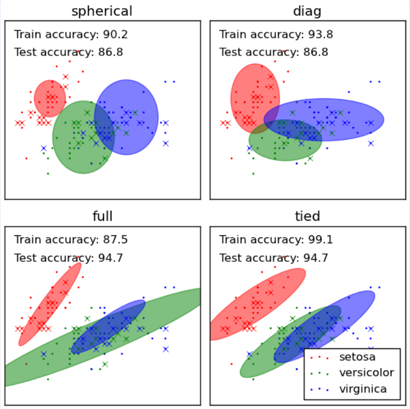
$$\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{\boldsymbol{x} \in \mathcal{D}_i} \boldsymbol{x}$$

- The sum-of-squared errors is defined as:

$$E = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \mathcal{D}_i} ||\boldsymbol{x} - \boldsymbol{\mu}_i||^2$$

- Measures the squared error incurred in representing each example with its cluster mean

Gaussian Mixture Model (GMM)



Setting

- Cluster examples using a mixture of Gaussian distributions
- Assume number of Gaussians is given
- Estimate mean and possibly variance of each Gaussian

Gaussian Mixture Model (GMM)

Parameter Estimation

- Maximum likelihood estimation cannot be applied as cluster assignment of examples is unknown
- Expectation-Maximization approach:
  1. Compute expected cluster assignment given current parameter setting
  2. Estimate parameters given cluster assignment
  3. Iterate

Example: estimating means of *k* univariate Gaussians

Setting

- A dataset of $x_1, \dots, x_n$ examples is observed
- For each example x_i, cluster assignment is modelled as $z_{i1}, \dots, z_{ik}$ binary latent (i.e. unknown) variables
- $z_{ij} = 1$ if Gaussian j generated x_i, 0 otherwise.
- Parameters to be estimated are the $\mu_1, \dots, \mu_k$ Gaussians means
- All Gaussians are assumed to have the same (known) variance σ^2

Example: estimating means of k univariate Gaussians

Algorithm

1. Initialize $h = \langle \mu_1, \ldots, \mu_k \rangle$
2. Iterate until difference in maximum likelihood (ML) is below a certain threshold:

   E-step Calculate expected value $\mathrm{E}[z_{ij}]$ of each latent variable assuming current hypothesis $h = \langle \mu_1, \ldots, \mu_k \rangle$ holds

   M-step Calculate a new ML hypothesis $h' = \langle \mu'_1, \ldots, \mu'_k \rangle$ assuming values of latent variables are their expected values just computed. Replace $h \leftarrow h'$

Example: estimating means of *k* univariate Gaussians

Algorithm

E-step The expected value of z_{ij} is the probability that x_i is generated by Gaussian j assuming hypothesis $h = \langle \mu_1, \ldots, \mu_k \rangle$ holds:

$$\mathrm{E}[z_{ij}] = \frac{p(x_i|\mu_j)}{\sum_{l=1}^{k} p(x_i|\mu_l)} = \frac{\exp -\frac{1}{2\sigma^2}(x_i - \mu_j)^2}{\sum_{l=1}^{k} \exp -\frac{1}{2\sigma^2}(x_i - \mu_l)^2}$$

M-step The maximum-likelihood mean μ_j is the weighted sample mean, each instance being weighted by its probability of being generated by Gaussian j:

$$\mu'_j = \frac{\sum_{i=1}^{n} E[z_{ij}] x_i}{\sum_{i=1}^{n} E[z_{ij}]}$$

Expectation-Maximization (EM)

Formal setting

- We are given a dataset made of an observed part X and an unobserved part Z
- We wish to estimate the hypothesis maximizing the expected log-likelihood for the data, with expectation taken over unobserved data:

$$h^* = \mathrm{argmax}_h \mathrm{E}_Z[ln\, p(X, Z|h)]$$

Problem

The unobserved data Z should be treated as random variables governed by the distribution depending on X and h

Expectation-Maximization (EM)

Generic algorithm

1. Initialize hypothesis h
2. Iterate until convergence

E-step Compute the expected likelihood of an hypothesis h' for the full data, where the unobserved data distribution is modelled according to the current hypothesis h and the observed data:

$$Q(h'; h) = \mathrm{E}_Z[\ln p(X, Z|h')|h, X]$$

M-step replace the current hypothesis with the one maximizing $Q(h'; h)$

$$h \leftarrow \mathrm{argmax}_{h'} Q(h'; h)$$

Example: estimating means of k univariate Gaussians

Derivation

- the likelihood of an example is:

$$p(x_i, z_{i1}, \ldots, z_{ik}|h') = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\sum_{j=1}^{k} z_{ij} \frac{(x_i - \mu'_j)^2}{2\sigma^2}\right]$$

- the dataset log-likelihood is:

$$\ln p(X, Z|h') = \sum_{i=1}^{n} \left(\ln \frac{1}{\sqrt{2\pi}\sigma} - \sum_{j=1}^{k} z_{ij} \frac{(x_i - \mu'_j)^2}{2\sigma^2}\right)$$

Example: estimating means of *k* univariate Gaussians

E-step

- the expected log-likelihood (remember linearity of the expectation operator):

$$E_Z[\ln p(X, Z|h')] = \mathrm{E}_Z\left[\sum_{i=1}^{n}\left(\ln\frac{1}{\sqrt{2\pi}\sigma} - \sum_{j=1}^{k} z_{ij}\frac{(x_i - \mu_j')^2}{2\sigma^2}\right)\right]$$

$$= \sum_{i=1}^{n}\left(\ln\frac{1}{\sqrt{2\pi}\sigma} - \sum_{j=1}^{k} E[z_{ij}]\frac{(x_i - \mu_j')^2}{2\sigma^2}\right)$$

- The expectation given current hypothesis *h* and observed data *X* is computed as:

$$\mathrm{E}[z_{ij}] = \frac{p(x_i|\mu_j)}{\sum_{l=1}^{k} p(x_i|\mu_l)} = \frac{\exp -\frac{1}{2\sigma^2}(x_i - \mu_j)^2}{\sum_{l=1}^{k} \exp -\frac{1}{2\sigma^2}(x_i - \mu_l)^2}$$

Example: estimating means of *k* univariate Gaussians

M-step

- The likelihood maximization gives:

$$\mathrm{argmax}_{h'} Q(h'; h) = \mathrm{argmax}_{h'} \sum_{i=1}^{n} \left(\ln \frac{1}{\sqrt{2\pi\sigma}} - \sum_{j=1}^{k} E[z_{ij}] \frac{(x_i - \mu'_j)^2}{2\sigma^2} \right)$$

$$= \mathrm{argmin}_{h'} \sum_{i=1}^{n} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2$$

- zeroing the derivative wrt to each mean we get:

$$\frac{\partial}{\partial \mu'_j} = -2 \sum_{i=1}^{n} E[z_{ij}](x_i - \mu'_j) = 0$$

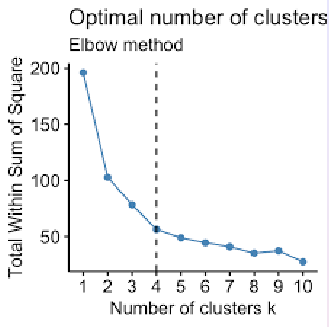
$$\mu'_j = \frac{\sum_{i=1}^{n} E[z_{ij}] x_i}{\sum_{i=1}^{n} E[z_{ij}]}$$

How to choose the number of clusters?

Elbow method: idea

- Increasing number of clusters allows for better modeling of data
- Needs to trade-off quality of clusters with quantity
- Stop increasing number of clusters when advantage is limited

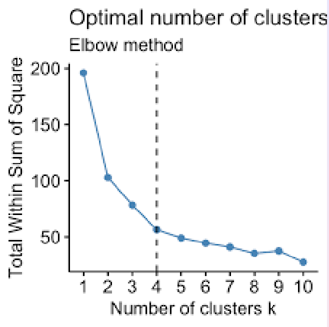
How to choose the number of clusters?



Elbow method: approach

1. Run clustering algorithm for increasing number of clusters
2. Plot clustering evaluation metric (e.g. sum of squared errors) for different *k*
3. Choose *k* when there is an angle (making an elbow) in the plot (drop in gain)

How to choose the number of clusters?



Elbow method: problem

The Elbow method can be ambiguous, with multiple candidate points (e.g. k=2 and k=4 in the figure).

How to choose the number of clusters?

Average silhouette method: idea

- Increasing the numbers of clusters makes **each cluster** more homogeneuous
- Increasing the number of clusters can make **different clusters** more similar
- Use quality metric that trades-off intra-cluster similarity and inter-cluster dissimilarity

How to choose the number of clusters?

Silhouette coefficient for example *i*

1. Compute the average dissimilarity between *i* and examples of its cluster *C*:

$$a_i = d(i, C) = \frac{1}{|C|} \sum_{j \in C} d(i, j)$$

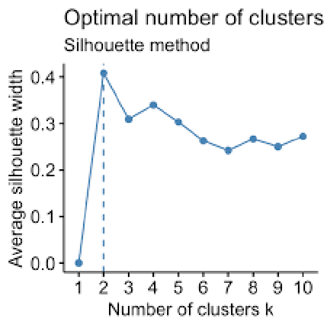
2. Compute the average dissimilarity between *i* and examples of each cluster $C' \neq C$, take the minimum:

$$b_i = \min_{C' \neq C} d(i, C')$$

3. The silhouette coefficient is:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

How to choose the number of clusters?



Average silhouette method: approach

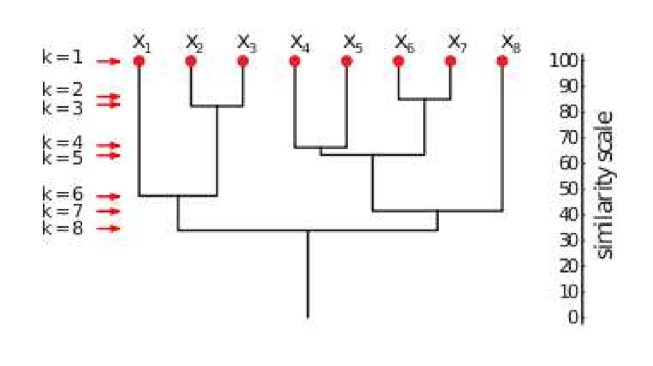
1. Run clustering algorithm for increasing number of clusters
2. Plot average (over examples) silhouette coefficient for different *k*
3. Choose *k* where the average silhouette coefficient is maximal

Hierarchical clustering

Setting

- Clustering does not need to be *flat*
- Natural grouping of data is often hierarchical (e.g. biological taxonomy, topic taxonomy, etc.)
- A hierarchy of clusters can be built on examples
- *Top-down* approach:
  - start from a single cluster with all examples
  - recursively split clusters into subclusters
- *Bottom-up* approach:
  - start with n clusters of individual examples (singletons)
  - recursively aggregate pairs of clusters

Dendograms


X_1
X_2
X_3
X_4
X_5
X_6
X_7
X_8
k = 1
k = 2
k = 3
k = 4
k = 5
k = 6
k = 7
k = 8
100
90
80
70
60
50
40
30
20
10
0
similarity scale

Agglomerative hierarchical clustering

Algorithm

1. Initialize:
    - Final cluster number k (e.g. k=1)
    - Initial cluster number $\hat{k} = n$
    - Initial clusters $\mathcal{D}_i = \{x_i\}, i \in 1, \dots, n$
2. while $\hat{k} > k$:
    1. find pairwise nearest clusters $\mathcal{D}_i, \mathcal{D}_j$
    2. merge $\mathcal{D}_i$ and $\mathcal{D}_j$
    3. update $\hat{k} = \hat{k} - 1$

Note

Stopping criterion can be threshold on pairwise similarity

Measuring cluster similarities

Similarity measures

- Nearest-neighbour

$$d_{min}(\mathcal{D}_i, \mathcal{D}_j) = \min_{\boldsymbol{x} \in \mathcal{D}_i, \boldsymbol{x}' \in \mathcal{D}_j} ||\boldsymbol{x} - \boldsymbol{x}'||$$

- Farthest-neighbour

$$d_{max}(\mathcal{D}_i, \mathcal{D}_j) = \max_{\boldsymbol{x} \in \mathcal{D}_i, \boldsymbol{x}' \in \mathcal{D}_j} ||\boldsymbol{x} - \boldsymbol{x}'||$$

- Average distance

$$d_{avg}(\mathcal{D}_i, \mathcal{D}_j) = \frac{1}{n_i n_j} \sum_{\boldsymbol{x} \in \mathcal{D}_i} \sum_{\boldsymbol{x}' \in \mathcal{D}_j} ||\boldsymbol{x} - \boldsymbol{x}'||$$

- Distance between means

$$d_{mean}(\mathcal{D}_i, \mathcal{D}_j) = ||\boldsymbol{\mu}_i - \boldsymbol{\mu}_j||$$

- d_{min} and d_{max} are more sensitive to *outliers*

Stepwise optimal hierachical clustering

Algorithm

```
1 Initialize:
    • Final cluster number $k$ (e.g. k=1)
    • Initial cluster number $\hat{k} = n$
    • Initial clusters $\mathcal{D}_i = \{x_i\}, i \in 1, \ldots, n$
2 while $\hat{k} > k$:
    1 find best clusters $\mathcal{D}_i, \mathcal{D}_j$ to merge according to evaluation criterion
    2 merge $\mathcal{D}_i$ and $\mathcal{D}_j$
    3 update $\hat{k} = \hat{k} - 1$
```

References

- R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification (2nd edition)*, Wiley-Interscience, 2001 (chapter 10)