CS-EJ3211 Machine Learning with Python

Session 5 - Clustering

Shamsi Abdurakhmanova

Aalto University
FITech

10.03.22

Unsupervised learning

- Data without labels
- Used to:
    - find new data representation, which is "easier" to interpret
    - prepare/transform data before applying supervised learning algorithms
- Examples:
    - dimensionality reduction
    - clustering

Clustering

Decompose dataset to subsets (subgroups) - **clusters**.

"Similar" datapoints are assigned to the same cluster.

Different clustering algorithms use different measures of similarity.

Examples:

market research (customer segmentation), recommendation systems, search result clustering, social network analysis.

Clustering

Clustering methods are roughly divided into two groups:

- **Hard clustering** methods - assign each data point to exactly one cluster
- **Soft clustering** methods - assign each data point to several different clusters with varying degrees of belonging

Hard Clustering: K-means

- Given: number of clusters k (hyperparameter)
- Similarity measure: Euclidean norm (distance)

Hard Clustering: K-means

Algorithm:

- randomly select k samples as initial centroids
- while true:
  - create k clusters by assigning each sample to the closest centroid

$$\hat{y}^{(i)} = \underset{c \in \{1,\ldots,k\}}{\operatorname{argmin}} \|\mathbf{x}^{(i)} - \mu^{(c)}\|^2$$

  - create k new centroids by averaging samples in each cluster
  - if centroids do not change (algorithm converged):
    break

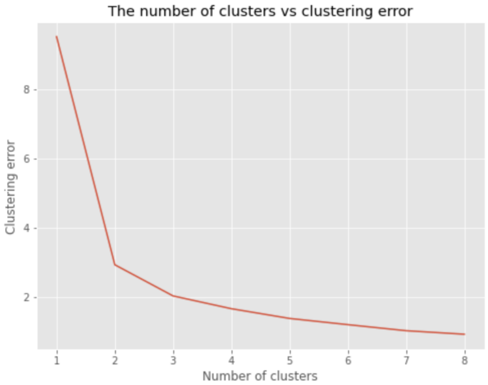
Animation

K-means Clustering

- Given enough time, K-means will always converge. However this may be to a local minimum (dependent on the initialization of the centroids)
- $\rightarrow$ Do computation several times, with different initializations of the centroids
- sklearn.cluster.KMeans has default param `init='k-means++'`. This initializes the centroids to be (generally) distant from each other

K-means: How many clusters?

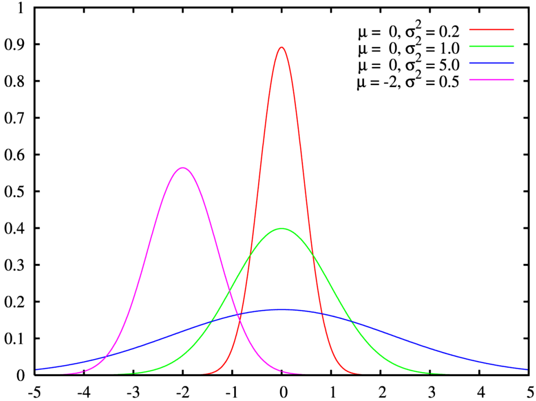
- Visualization - few clusters
- Pre-processing before supervised methods - use validation set to choose n.o. clusters
- "Elbow" method



Soft clustering - Gaussian Mixture Models

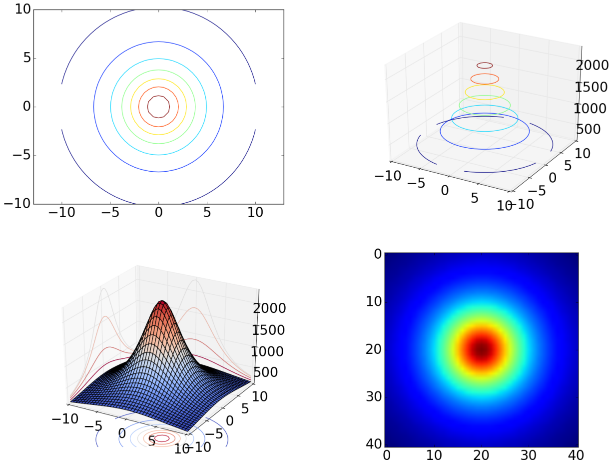
Gaussian probability distribution (1D):

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$



Soft clustering - Gaussian Mixture Models

Gaussian probability distribution (2D, bivariate):



Soft clustering - Gaussian Mixture Models

- Data is assumed to be drawn from k different multivariate Gaussian distributions
- Each Gaussian distributions is parametrized by a mean vector $\mu^{(c)}$ and a covariance matrix $\mathbf{C}^{(c)}$
- The model has the parameters p_c representing the probability of drawing a data point from the distribution c
- The model is fitted by finding the parameters $\mu_c, \mathbf{C}_c, p_c$, for each $c = 1, \ldots, k$ (where k is the number of clusters), that maximize the likelihood of the observed data.

Soft clustering - Gaussian Mixture Models

Algorithm:

- randomly select Gaussian parameters $\mu^{(c)}$, $\mathbf{C}^{(c)}$
- while true:
  - compute probabilities of a datapoint coming from each Gaussian

$$\mathbf{y}_c^{(i)} = \frac{p_c \mathcal{N}(\mathbf{x}^{(i)}; \mu^{(c)}, \mathbf{C}^{(c)})}{\sum_{c'=1}^{k} p_{c'} \mathcal{N}(\mathbf{x}^{(i)}; \mu^{(c')}, \mathbf{C}^{(c')})}$$

  - update parameters $\mu^{(c)}$, $\mathbf{C}^{(c)}$ to maximize likelihood
  - if log-likelihood do not change significantly (algorithm converged): break

Animation

additional material: EM, GMM lecture

Clustering with sklearn

Clustering with sklearn