Introduction to Data Engineering
07 Clustering, Outlier, Novelty Detection and Duplicate Detection

Junbo Huang and Angelie Kraft and Cedric Möller and David Rath and Ricardo Usbeck

https://lernen.min.uni-hamburg.de/course/view.php?id=2917

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?
- Behaviorists: conditioning -> Reinforcement learning

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?
- Behaviorists: conditioning -> Reinforcement learning
- Albert Bandura: social learning theory -> Imitation learning

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?
- Behaviorists: conditioning -> Reinforcement learning
- Albert Bandura: social learning theory -> Imitation learning
- Lower-level learning?
  - Perception? Cognition? Concept learning?

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?
- Behaviorists: conditioning -> Reinforcement learning
- Albert Bandura: social learning theory -> Imitation learning
- Lower-level learning?
  - Perception? Cognition? Concept learning?
  - Wittgenstein, 1953: language acquisition through pointing[1]

[1] Wittgenstein, L. (1953). Philosophical investigations.

Intuition: Aspect of Human Learning

- How do we learn? -> how do children learn?
- Behaviorists: conditioning -> Reinforcement learning
- Albert Bandura: social learning theory -> Imitation learning
- Lower-level learning?
  - Perception? Cognition? Concept learning?
  - Wittgenstein, 1953: language acquisition through pointing[2].
  - Eleanor Rosch, 1978: Category formation is strongly connected to forming prototypical concepts[3].

---

[2] Wittgenstein, L. (1953). Philosophical investigations.
[3] Rosch, E. (1978). Principles of categorization.

Intuition: Aspect of Human Learning

- The concept of birdiness.
- Robin is a more typical bird than a penguin.

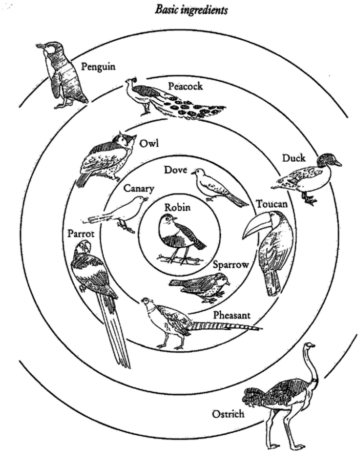


Figure 1 Birdiness rankings

Source: Aitchison, J. (1994). Words in the mind: An introduction to the mental lexicon. page 54.

Intuition: Aspect of Human Learning

- The concept of birdiness.
- Robin is a more typical bird than a penguin.
- Bias in prototyping?
  - Misunderstanding.

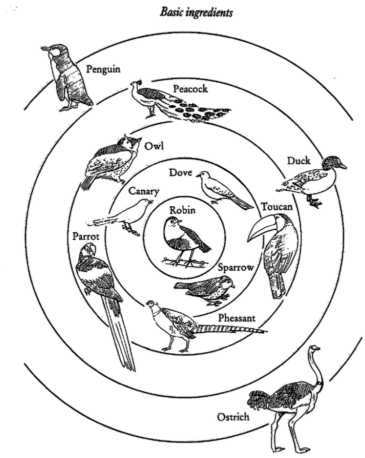


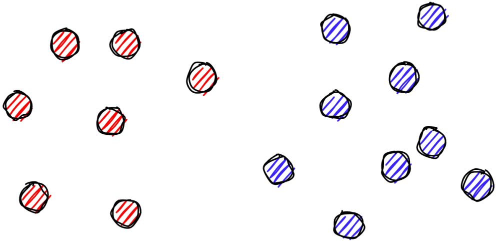
Figure 1 Birdiness rankings

Source: Aitchison, J. (1994). Words in the mind: An introduction to the mental lexicon. page 54.

Overview

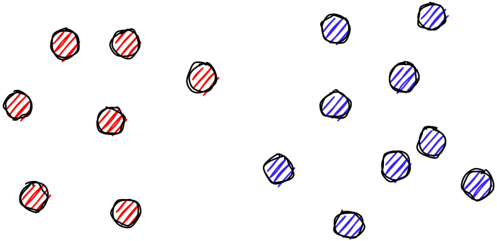
Two Types of Distance Measures

What are the two types of distance measures?



Two Types of Distance Measures

- Distance between points
- Distance between clusters



Distance Metric

Consider a metric space $\mathcal{X}$. A distance metric is a mapping $d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \inf)$ which satisfies the following properties:

- non-negativity: $d(x_i, x_j) \geq 0$
- identity: $d(x_i, x_j) = 0 \Leftrightarrow x_i = x_j$
- symmetry: $d(x_i, x_j) = d(x_j, x_i)$
- triangle inequality: $d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j)$

where $x_i, x_j, x_k \in \mathcal{X}$.

Distance Metric

Remarks:

- In Mathematics, the term metric is used only when the axioms are fulfilled.
- In ML, distance metric often refers to the similarity or dissimilarity measure and it may not satisfy all the axioms (e.g., cosine distance).
- Therefore, if you are clear that the axioms are satisfied, use the term metric.

Distance Between Points

Given two points $x, y \in \mathbb{R}^n$,

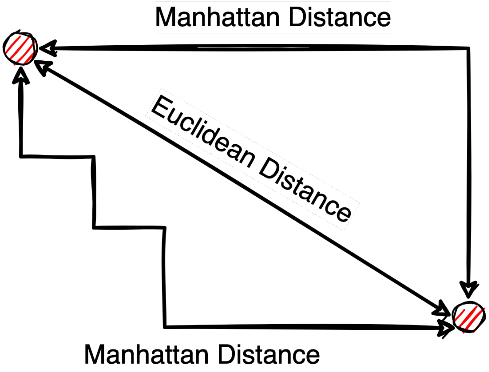
- Euclidean Distance:

  $$d(x, y) = \sqrt{\sum_i^n (x_i - y_i)^2}$$

- Manhattan Distance / City Block Distance:

  $$d(x, y) = \sum_i^n |x_i - y_i|$$

- (Any parametrized distance metric)



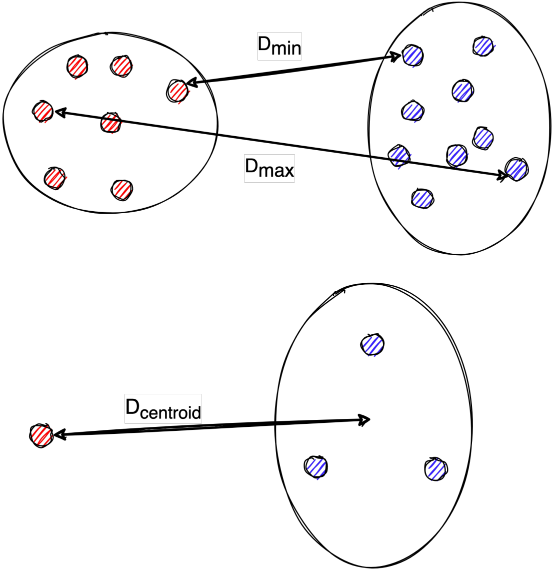
Distance Between Clusters

How can one measure distance between two clusters?

Distance Between Clusters

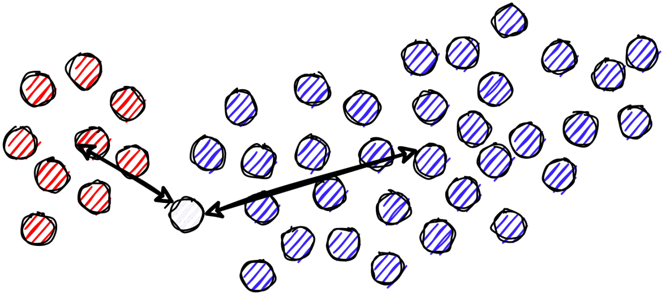
Given two clusters X and Y:

- $D_{min}(X, Y) = \min_{x \in X, y \in Y} d(x, y)$
- $D_{max}(X, Y) = \max_{x \in X, y \in Y} d(x, y)$
- $D_{mean}(X, Y) = \frac{1}{|X||Y|} \sum_{x \in X, y \in Y} d(x, y)$
- $D_{centroid}(X, Y) = d(\frac{1}{|X|} \sum_{x \in X} x, \frac{1}{|Y|} \sum_{y \in Y} y)$



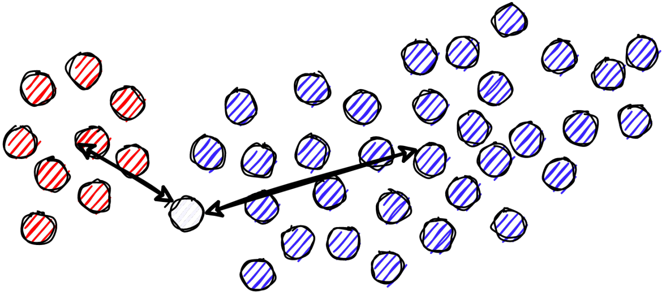
Distance Between Clusters

$D_{min}, D_{max}, D_{mean}, D_{centroid}$ ignore the variances of data.



Distance Between Clusters

D_{min}, D_{max}, D_{mean}, $D_{centroid}$ ignore the variances of data.



- Solution: Mahalanobis Distance

Mahalanobis Distance

Idea: Scaling of distances using the covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$.

$$D_\Sigma(X, Y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y))}$$

Properties:

- Σ characterizes the distribution of data.
- If Σ is an identity matrix, the above equation represents Euclidean distance.
- It is a good distance measure between sets/clusters.
- Can also be used to measure distances between a point and sets/clusters.
- Spectral distance metric learning: parametrizing Σ.

A Clustering Problem

Clustering German cities based on the location using Euclidean distance.

- Hamburg
- Bremen
- Hannover
- Leipzig
- Frankfurt
- Nürnberg

Source: Google maps

A Clustering Problem

$$\mathbf{x} = \begin{bmatrix} 53.55 & 9.99 \\ 53.07 & 8.8 \\ 52.37 & 9.73 \\ 51.33 & 12.37 \\ 50.13 & 8.66 \\ 49.45 & 11.07 \end{bmatrix}$$

Or, a $n \times d$ matrix (n cities and d features)

Distance Matrix

A matrix where each entry represents the euclidean distance (in km) between two cities.

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

Types of Clustering Algorithm

- Hard clustering algorithms
  - Hard assignment of clusters
  - Hierarchical clustering, k-means[4]...
- Soft clustering algorithms
  - Probabilistic assignment of clusters
  - Gaussian Mixture Models...

[4] https://colab.research.google.com/drive/1x95aarcKwFMs26u4Fwn5rQThWINHhQ65?usp=sharing

Hierarchical Clustering

- Do not need to pre-define number of clusters k.
- Two complementary methods:
  - Agglomerative Clustering: bottom-up
  - Divisive Clustering: top-down

Agglomerative Clustering

Basic agglomerative clustering:

1. Assign each object to its own single-object cluster.
2. Choose the closest pair of clusters and merge them into a single cluster.
3. Calculate the distance between the new cluster and each of the old clusters.
4. Assign Repeat steps 2 and 3 until all the objects are in a single cluster.

Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}

Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}

Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
3. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}

Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
3. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
4. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt, Nürnberg}

Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
3. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
4. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt, Nürnberg}
5. {Hamburg, Bremen, Hannover, Leipzig}, {Frankfurt, Nürnberg}

Agglomerative Clustering

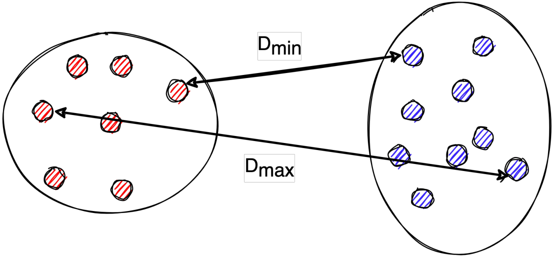
	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
3. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
4. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt, Nürnberg}
5. {Hamburg, Bremen, Hannover, Leipzig}, {Frankfurt, Nürnberg} -> Single-linkage clustering
5. {Hamburg, Bremen, Hannover}, {Leipzig, Frankfurt, Nürnberg} -> Complete-linkage clustering

Recall: Distance Between Clusters

Given two clusters X and Y:

- $D_{min}(X, Y) = \min_{x \in X, y \in Y} d(x, y)$
  - single-linkage clustering
- $D_{max}(X, Y) = \max_{x \in X, y \in Y} d(x, y)$
  - complete-linkage clustering
- $D_{mean}(X, Y) = \frac{1}{|X||Y|} \sum_{x \in X, y \in Y} d(x, y)$
- $D_{centroid}(X, Y) = d(\frac{1}{|X|} \sum_{x \in X} x, \frac{1}{|Y|} \sum_{y \in Y} y)$



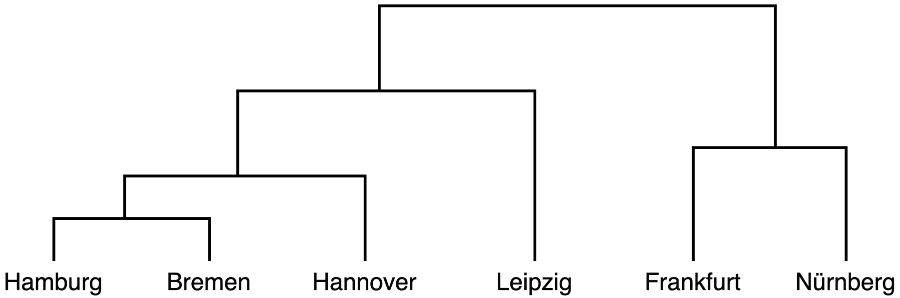
Agglomerative Clustering

	Hamburg	Bremen	Hannover	Leipzig	Frankfurt	Nürnberg
Hamburg	0	95	133	294	393	462
Bremen		0	100	310	330	433
Hannover			0	214	262	338
Leipzig				0	293	229
Frankfurt					0	187
Nürnberg						0

1. {Hamburg}, {Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
2. {Hamburg, Bremen}, {Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
3. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt}, {Nürnberg}
4. {Hamburg, Bremen, Hannover}, {Leipzig}, {Frankfurt, Nürnberg}
5. {Hamburg, Bremen, Hannover, Leipzig}, {Frankfurt, Nürnberg}
6. {Hamburg, Bremen, Hannover, Leipzig, Frankfurt, Nürnberg}

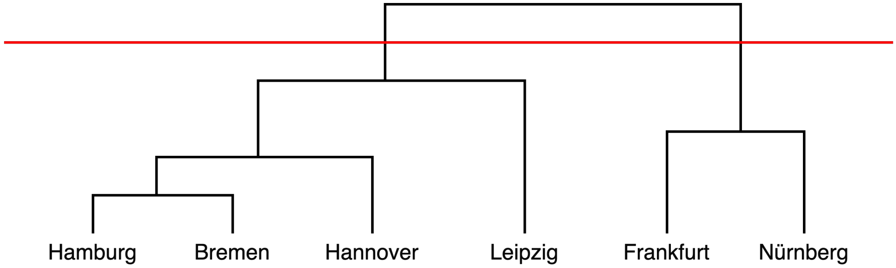
Agglomerative Clustering

Dendrogram: a diagram representing a binary tree.



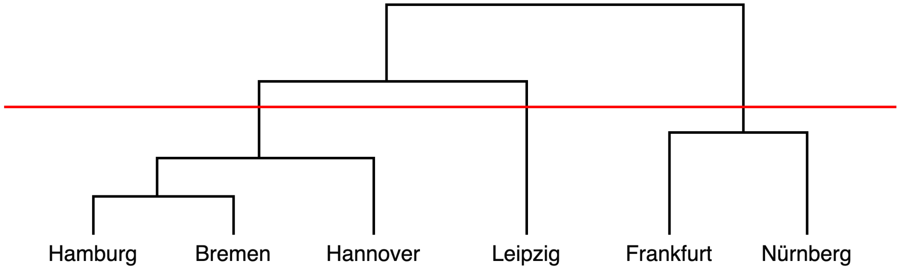
Agglomerative Clustering

Dendrogram: a diagram representing a binary tree.



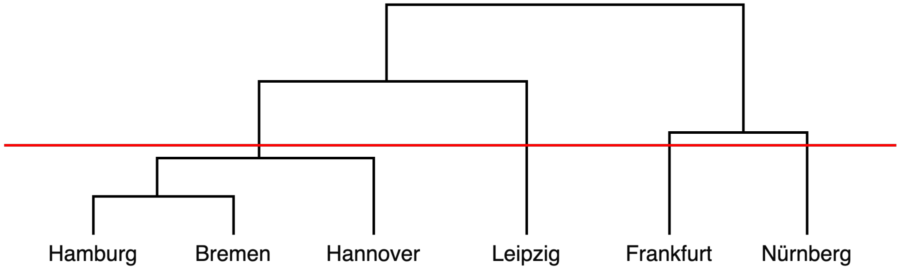
Agglomerative Clustering

Dendrogram: a diagram representing a binary tree.



Agglomerative Clustering

Dendrogram: a diagram representing a binary tree.



Agglomerative Clustering

Properties of agglomerative clustering?

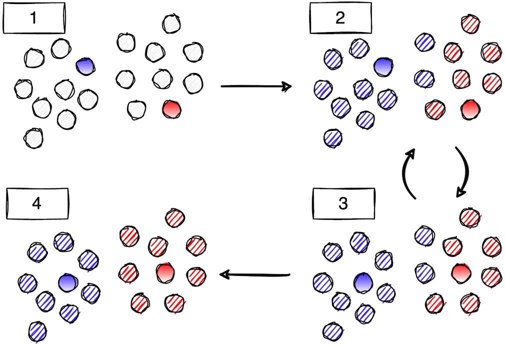
Agglomerative Clustering

Properties:

- Any distance measure can be used
- we only need the distance matrix
- No parameters
- Resulting dendrogram needs to be analyzed to decide number of desired clusters
- Slow when the number of samples is large
- Greedy, deterministic

K-means Algorithm

1. Randomly choose k examples as initial centroids.
2. Create k clusters by assigning each example to its closest centroid.
3. Compute k new centroids by averaging examples in each cluster.
4. Repeat step 2 and 3 until centroids don't change.



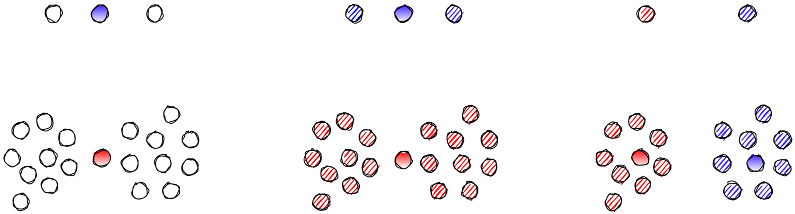
How to choose k?

Possible solutions:

- Run k-means multiple times with random initialization.
- Take a subset of the data, and run hierarchical clustering.
- Statistics
  - The elbow method
    - Calculate the Within-Cluster-Sum of Squared Errors (WCSS) for different values of k
  - The Silhouette Method
    - Measure how similar a point is to its own cluster (cohesion) compared to other clusters (separation)

Properties of K-means

- Only one parameter k.
  - Implicitly defines scale and resulting shape of clusters.
- Fast.
- Greedy, non-deterministic -> local optima, depending on initial conditions.



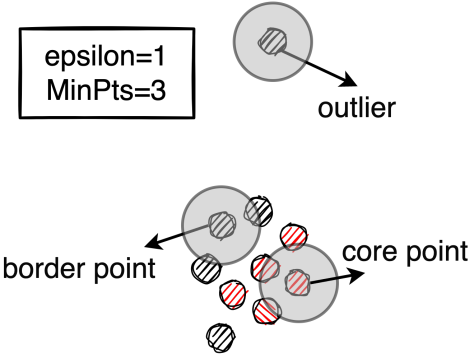
Initial Conditions — Resulting local optimum — Global optimum

Problems of K-means

- Sensitive to outliers.
  - Outliers highly influence clustering result in naïve k-means;
  - Outliers can be handled in the preprocessing step, or with algorithms which are robust to outliers,e.g., DBSCAN.
- What if we don't know k?
  - Other algorithms, e.g., hierarchical clustering or DBSCAN.
- What if data does not depict circular shape?
  - If you know how the data should look like,
  - You can choose other algorithms that have a more relaxed constraint on the shape of clusters, e.g., soft clustering or DBSCAN.

DBSCAN Clustering

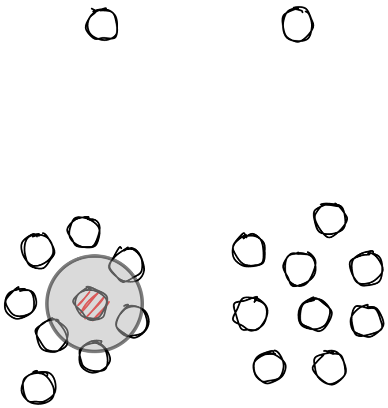
- Ester, 1996
- No need to define k
- Density-based clustering algorithm
  - Density connected points belong to the same cluster
- Not entirely deterministic.
- Two parameters:
  - epsilon ϵ
  - Minimum number of points MinPts



DBSCAN Clustering

Steps:

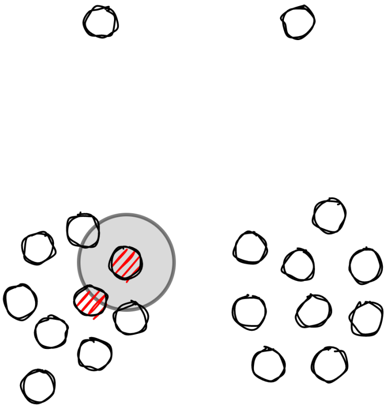
1. Find all core points given ϵ and *MinPts*.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.
4. Assign points that do not belong to any cluster as noise/ outliers.



DBSCAN Clustering

Steps:

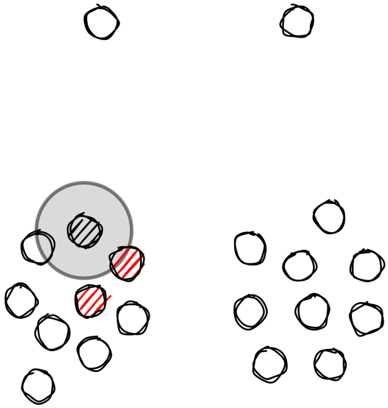
1. Find all core points given ϵ and *MinPts*.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.
4. Assign points that do not belong to any cluster as noise/ outliers.



DBSCAN Clustering

Steps:

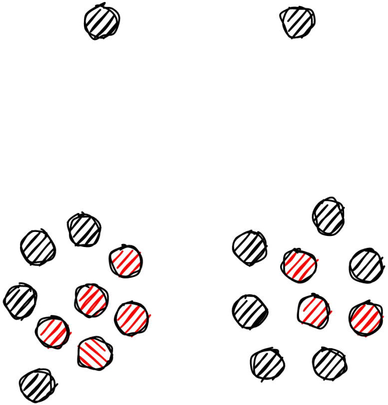
1. Find all core points given ϵ and *MinPts*.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.
4. Assign points that do not belong to any cluster as noise/ outliers.



DBSCAN Clustering

Steps:

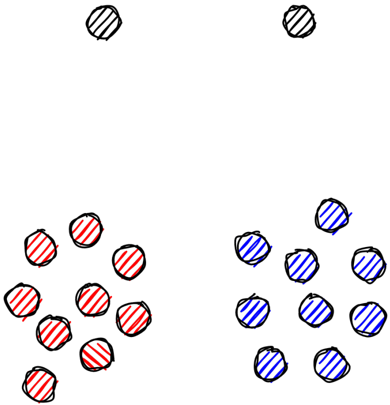
1. Find all core points given ϵ and *MinPts*.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.
4. Assign points that do not belong to any cluster as noise/ outliers.



DBSCAN Clustering

Steps:

1. Find all core points given ϵ and *MinPts*.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.
4. Assign points that do not belong to any cluster as noise/ outliers.



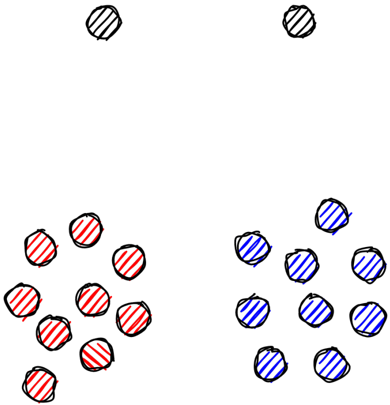
DBSCAN Clustering

Pros:

1. Robust to outliers.
2. No need to define k
3. Arbitrary cluster shape.

Cons:

1. Difficult to choose ϵ if data is not well understood.
2. "curse of dimensionality".
    1. a good distance measure is difficult.



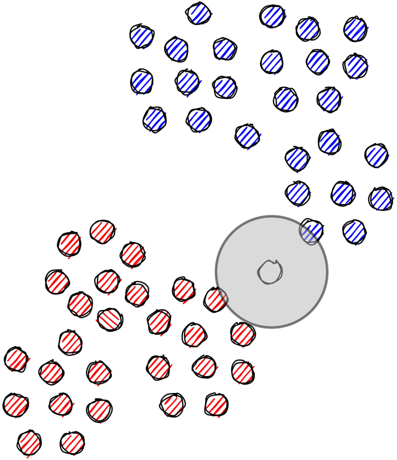
DBSCAN Clustering

Problems of DBSCAN?

DBSCAN Clustering

One problem:

- How to cluster a point between two clusters?
- Both k-means and DBSCAN assign one cluster to each data point.
- There is no "likelihood" in cluster assignments.



Soft Clustering

So far,

- clusters are described by a set of data points or by the centers
  -> clusters are disjoint
- This is called hard clustering.
- Each data point is assigned to a single cluster.
- No way to express uncertainty/confidence about the assignment to a cluster.

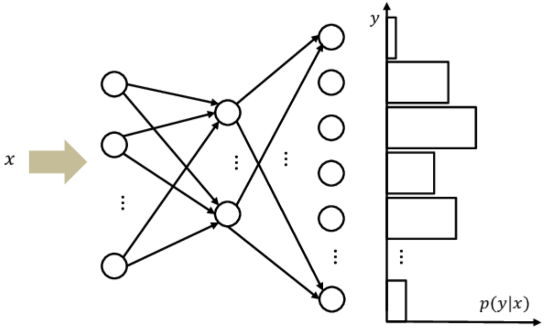
Now,

- Describe data by probability distribution $P(x)$.
- Soft clustering:
- A data point is assigned to the clusters with uncertainty/confidence.
- As a result, clusters do not have hard boundaries.

Soft clustering

Let's think about soft assignment in Deep Learning...

- What is the probability distribution outputted by a neural network?

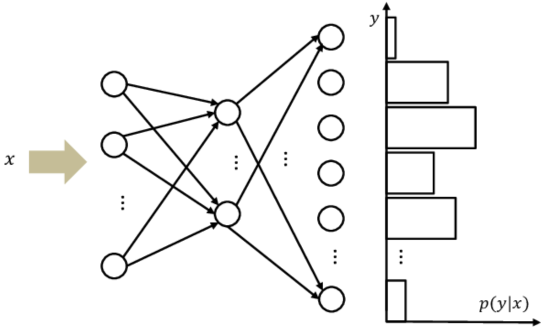


Source: Feron, B., Moraleda, G., Vossen, J., & Monti, A. Probabilistic Short-Term Residential Load Forecasting based on Feed Forward and LSTM Recurrent Neural Networks.

Soft clustering

Let's think about soft assignment in Deep Learning...

- What is the probability distribution outputted by a neural network?
- Softmax$(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$
- What is probability?

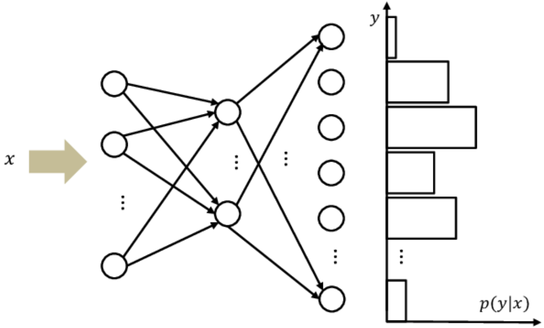


Source: Feron, B., Moraleda, G., Vossen, J., & Monti, A. Probabilistic Short-Term Residential Load Forecasting based on Feed Forward and LSTM Recurrent Neural Networks.

Soft clustering

Let's think about soft assignment in Deep Learning...

- What is the probability distribution outputted by a neural network?
- $\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$
- Probability$\neq$Softmax output[5]



Source: Feron, B., Moraleda, G., Vossen, J., & Monti, A. Probabilistic Short-Term Residential Load Forecasting based on Feed Forward and LSTM Recurrent Neural Networks.

[5] Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017, July). On calibration of modern neural networks. In International Conference on Machine Learning (pp. 1321-1330). PMLR.

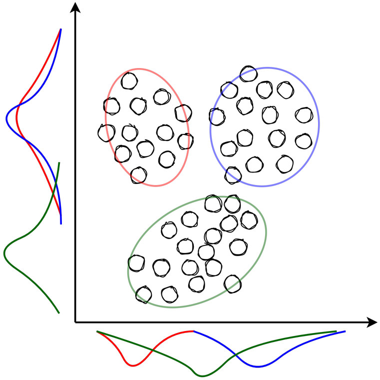
Soft clustering

Gaussian Mixture Models (GMMs)

- Assign a Gaussian to each cluster center.
- Linear superposition of K Gaussians.

$$p(x|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

- $\mathcal{N}(\cdot)$ denotes a Gaussian with mean μ and covariate matrix Σ.
- π_k denotes the *a priori* probability that a data point belongs to cluster k.



Gaussian Mixture Models

A density model where we combine a finite number of K Gaussian distributions $\mathcal{N} = (x|\mu_k, \Sigma_k)$ so that

$$p(x|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

$$0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

- $\theta := \{\pi_k, \mu_k, \Sigma_k : k = 1, 2, ..., K\}$
- Goal: optimize θ such that the GMM fits the data (Maximum Likelihood Estimate).

Gaussian Mixture Models

Problem formulation:

- Given dataset $\mathcal{X} = \{x_1, x_2, ..., x_N\}, n = 1, ..., N$, where x_n are drawn i.i.d. from an unknown distribution $p(x)$. Our objective is to find a good approximation of this unknown distribution $p(x)$ by means of a GMM with K mixture components, given a set of parameters $\theta := \{\pi_k, \mu_k, \Sigma_k : k = 1, 2, ..., K\}$.
- Typically done by likelihood maximization (or negative log likelihood minimization).

$$p(\mathcal{X}|\theta) = \prod_{n=1}^{N} p(x_n|\theta), \text{ where } p(x_n|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)$$

$$\mathcal{L} = -\log p(\mathcal{X}|\theta) = -\sum_{n=1}^{N} \log p(x_n|\theta) = -\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)$$

- Goal: find θ^* that minimizes the loss (negative log-likelihood) $\mathcal{L}$.

Gaussian Mixture Models

An important concept: *Responsibility*

$$r_{nk} := \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k))}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)}$$

.

- is defined as the *responsibility* of the k^{th} mixture component for data point x_n.
- *Responsibility* is the soft assignment of clusters.
- To update r_{nk}, we need $\{\pi_k, \mu_k, \Sigma_k : k = 1, 2, ..., K\}$.
- To update $\{\pi_k, \mu_k, \Sigma_k : k = 1, 2, ..., K\}$, we need r_{nk}.
- Needs an iterative solution.

Expectation-Maximization Algorithm

- Initialize π_k, μ_k, Σ_k.
- *E-step*: Evaluate responsibilities r_{nk} for every data point x_n using current parameters π_k, μ_k, Σ_k:

$$r_{nk} := \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k))}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}$$

- *M-step*: Re-estimate parameters π_k, μ_k, Σ_k using the current responsibilities r_{nk} (from E-step):

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n, \quad \pi_k = \frac{N_k}{N},$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^T.$$

- Repeat E-step and M-step until convergence.

Expectation-Maximization Algorithm

Mean updates μ_k

- The mean μ_k is pulled toward a data point x_n with the strength given by r_{nk}.

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n$$

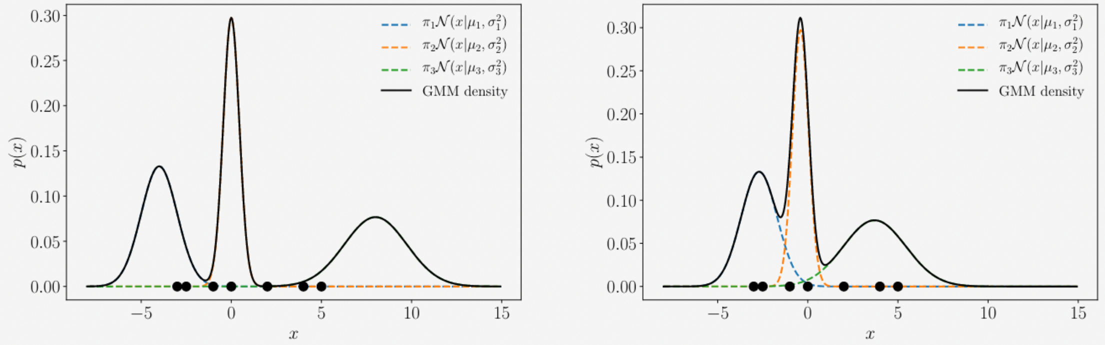
- N_k is defined as the total responsibility of the k^{th} mixture component for the entire dataset.

$$N_k := \sum_{n=1}^{N} r_{nk}$$

- Importance-weighted Monte Carlo estimate of the mean.

Expectation-Maximization Algorithm

Mean updates μ_k



(a) GMM density and individual components prior to updating the mean values.

(b) GMM density and individual components after updating the mean values.

Source: Mathematics for Machine Learning. page 355.

Expectation-Maximization Algorithm

Covariance updates Σ_k

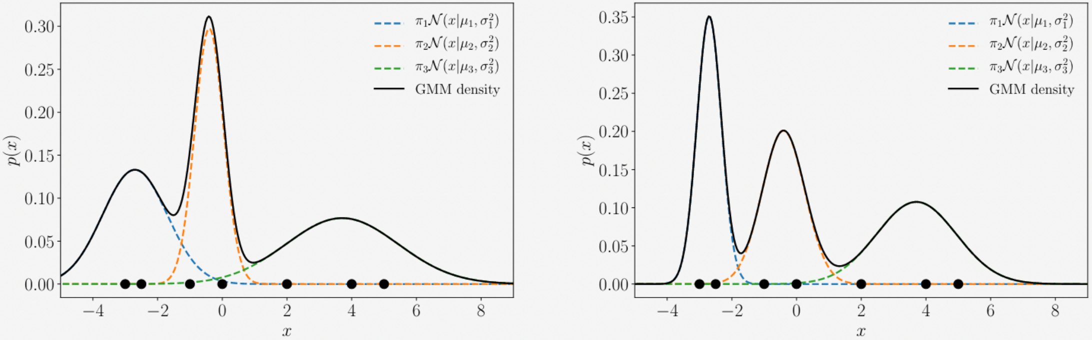
- The covariance matrix is re-estimated based on the new responsibilities r_k

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk}(x_n - \mu_k)(x_n - \mu_k)^T$$

- Importance-weighted covariance of data points x_n associated with the k^{th} mixture component.
- Proof:
  - Take the partial derivative of the log-likelihood with respect to Σ_k.
  - Set the partial derivative to 0.

Expectation-Maximization Algorithm

Covariance updates Σ_k



(a) GMM density and individual components prior to updating the variances.

(b) GMM density and individual components after updating the variances.

Source: Mathematics for Machine Learning. page 358.

Expectation-Maximization Algorithm

Mixture weights updates π_k

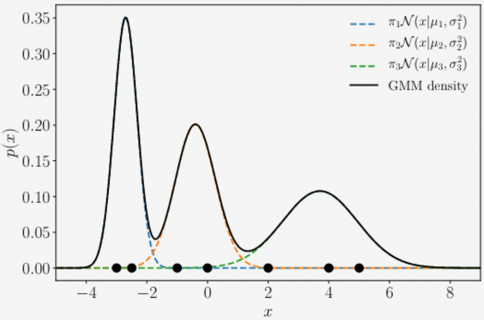
- The mixture weights are updated based on the new responsibilities r_k

$$\pi_k = \frac{N_k}{N}$$

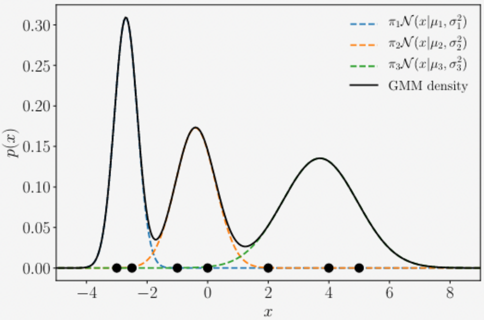
- N denotes number of data points.
- The ratio of the total responsibility of the k^{th} cluster and the number of data points.

Expectation-Maximization Algorithm

Mixture weights updates π_k



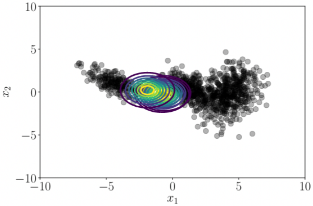
(a) GMM density and individual components prior to updating the mixture weights.



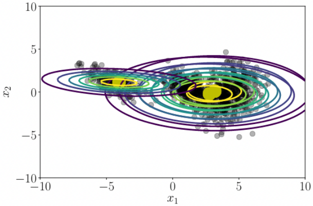
(b) GMM density and individual components after updating the mixture weights.

Source: Mathematics for Machine Learning. page 360.

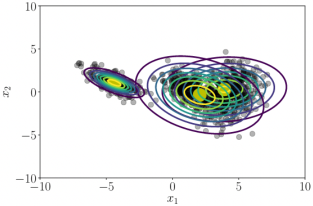
Expectation-Maximization Algorithm



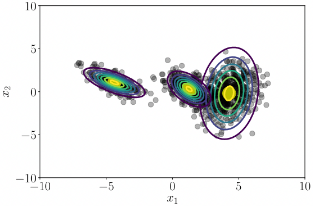
(c) EM initialization.



(d) EM after one iteration.



(e) EM after 10 iterations.



(f) EM after 62 iterations.

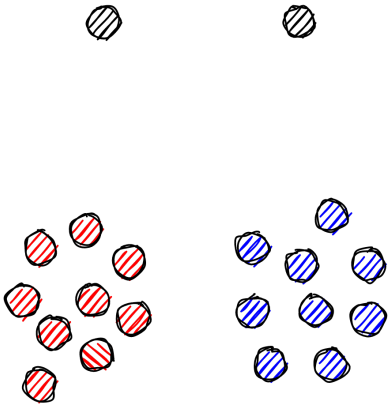
Source: Mathematics for Machine Learning. page 362.

New stuff?

- DeepCluster (Caron et al. 2018)
- Deep k-means (Fard, Thonet, and Gaussier 2020)
- Swapping Assignments between multiple Views (SwAV) (Caron et al. 2020)
- Prototypical contrastive learning (Li et al. 2020)

Outliers, Novelty and Duplicate Detection

- Anomaly detection
  - Outlier detection,
  - Novelty detection.
- Samples located in low-density region.
- Outliers typically pollutes the data,
- whereas novelties gain insight about the data.
- Methods:
  - DBSCAN,
  - Mahalanobis distance, etc.



Summary

- Different types of distance metrics and their characteristics.
  - Distance metrics between points,
  - Distance metrics between clusters.
- Hard clustering algorithms.
  - Hierarchical clustering, k-means, and DBSCAN.
  - Easy to use,
  - But no confidence measurement; can be slow due to "curse of dimensionality".
- Soft clustering algorithms.
  - Gaussian mixture models, Expectation-Maximization algorithm.
  - Soft assignment of clusters with uncertainty.
- Application: anomaly detection.

For next time, please read about data quality `https://5stardata.info/en/`

References

- Ester, M., Kriegel, H. P., Sander, J., Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).
- Deisenroth, M. P., Faisal, A. A., Ong, C. S. (2020). Mathematics for machine learning. Cambridge University Press.
- Ozdemir, S. (2016). Principles of data science. Packt Publishing Ltd.
- Ghojogh, B., Ghodsi, A., Karray, F., Crowley, M. (2022). Spectral, Probabilistic, and Deep Metric Learning: Tutorial and Survey. arXiv preprint arXiv:2201.09267.
- Caron, M., Bojanowski, P., Joulin, A., Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In Proceedings of the European conference on computer vision (ECCV) (pp. 132-149).

References

- Fard, M. M., Thonet, T., Gaussier, E. (2020). Deep k-means: Jointly clustering with k-means and learning representations. Pattern Recognition Letters, 138, 185-192.
- Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. Advances in Neural Information Processing Systems, 33, 9912-9924.
- la Rosa, L. E. C., Oliveira, D. A. B. (2022). Learning from Label Proportions with Prototypical Contrastive Clustering.
- Li, J., Zhou, P., Xiong, C., Hoi, S. C. (2020). Prototypical contrastive learning of unsupervised representations. arXiv preprint arXiv:2005.04966.
- Guo, C., Pleiss, G., Sun, Y., Weinberger, K. Q. (2017, July). On calibration of modern neural networks. In International Conference on Machine Learning (pp. 1321-1330). PMLR.