Data Science Academy

Data Science for Business – Becoming a Data Science Expert (D)

Pilot Presentation:
for participants of and use in the pilot only

06.09.2021

Boehringer Ingelheim

Agenda week two

	Introduction
1	Recap Basic Machine Learning and Python
2	Complex Models
3	Model Evaluation
4	Hyperparameters
5	**Unsupervised Learning**
6	**Gradient Descent**
7	Deep Learning and Image Recognition
8	Deep Learning and Natural Language Processing
9	Repetition
10	Bias and Ethics in Machine Learning
11	Introduction to Data Science with AWS

Schedule week two

Week 2			
	Day 1 **Monday, 06.09.2021**		**Day 2** **Tuesday, 07.09.2021**
Start: 12:00	Recap	Start: 12:00	Recap
	5 – Unsupervised Learning (Part 1)		6 – Gradient Descent (Part 1)
14:00 – 15:00	Break	14:00 – 15:00	Break
	5 – Unsupervised Learning (Part 2)		6 – Gradient Descent (Part 2)
End: 18:00	Q&A and Feedback	End: 18:00	Q&A and Feedback

We will also have several short coffee breaks in between.

Feedback for pilot training

We aim to provide a great training experience for you and are looking forward to receiving your feedback!

You will have three different ways to give us your feedback on each training day:

1. We will have an **anonymized** feedback collection **after the last session** of each day per **Myforms.**
2. We will have an **open feedback round and discussion** at the **end of each training day.**
3. Please also **take notes** regarding your ideas during the sessions: **locally or via the Mural Board** which you can reach via LINK.

Quiz: Recap week one

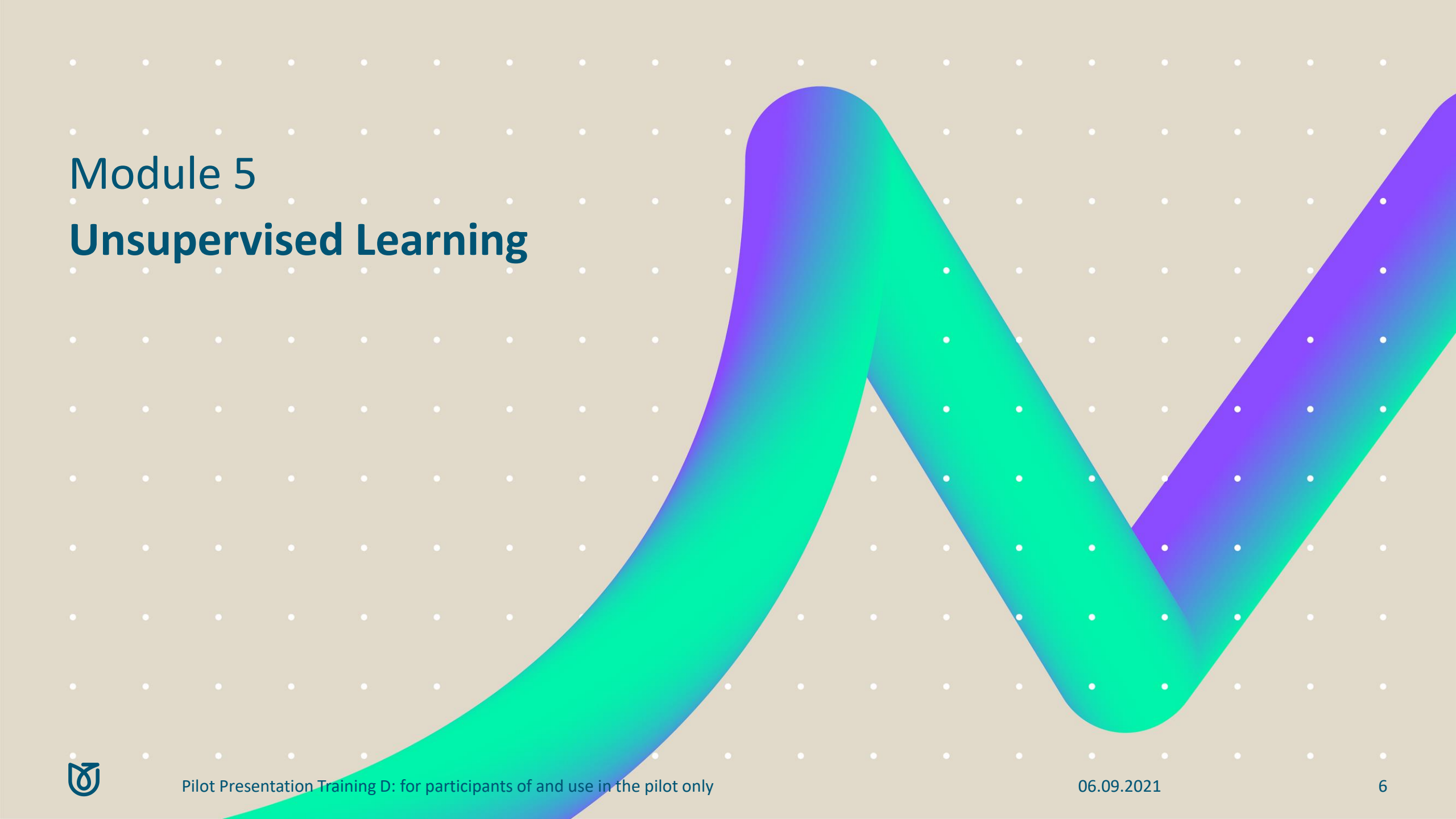
Please join at slido.com with #031 077.

Let’s go through some questions together.

Let’s see what you think. All answers will be anonymous.

Module 5

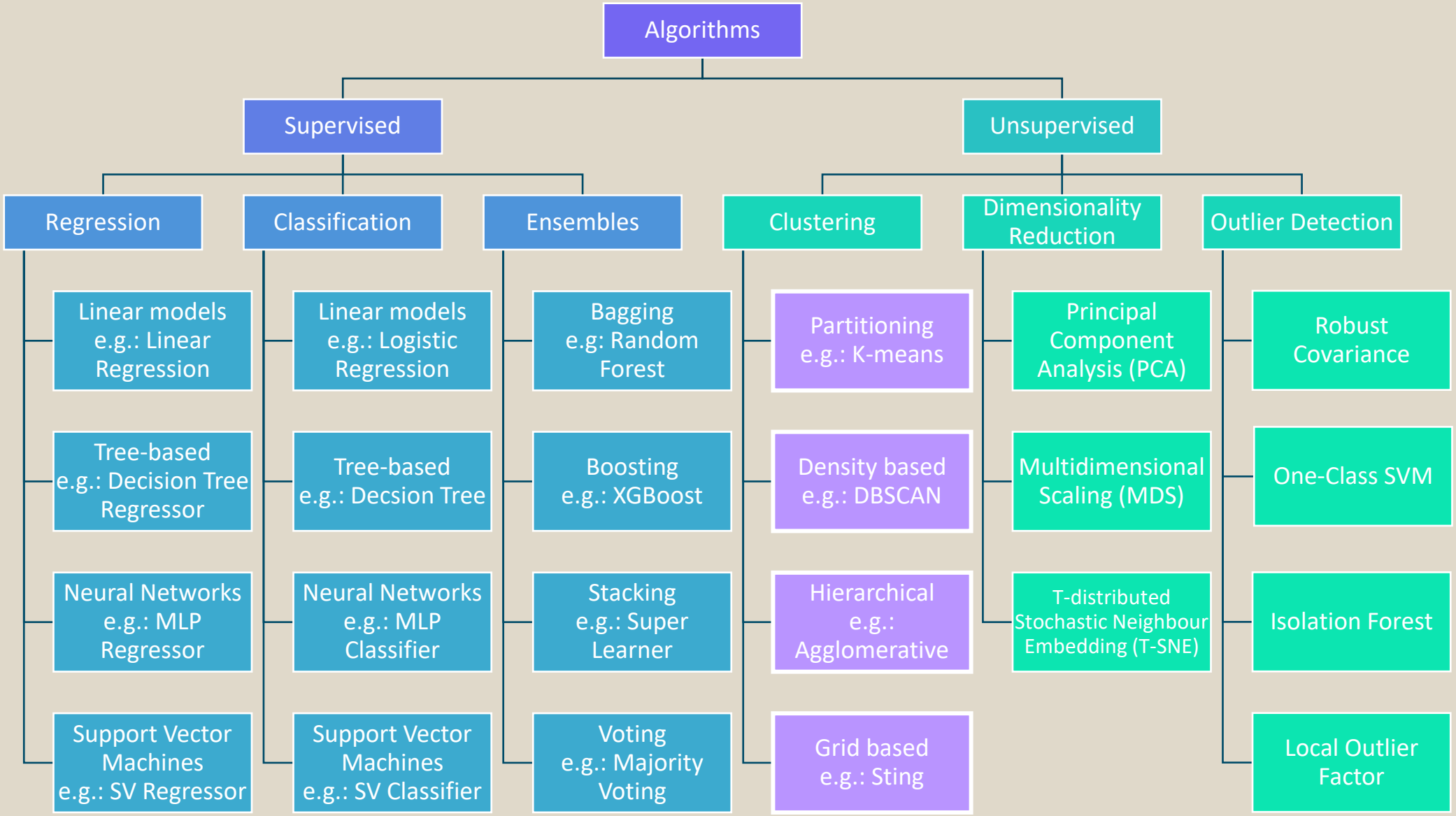
Unsupervised Learning



Agenda week one

	Introduction
1	Recap Basic Machine Learning and Python
2	Complex Models
3	Model Evaluation
4	Hyperparameters
5	**Unsupervised Learning**
6	Gradient Descent
7	Deep Learning and Image Recognition
8	Deep Learning and Natural Language Processing
9	Repetition
10	Bias and Ethics in Machine Learning
11	Introduction to Data Science with AWS

Selected model types



Introduction to clustering

Goal

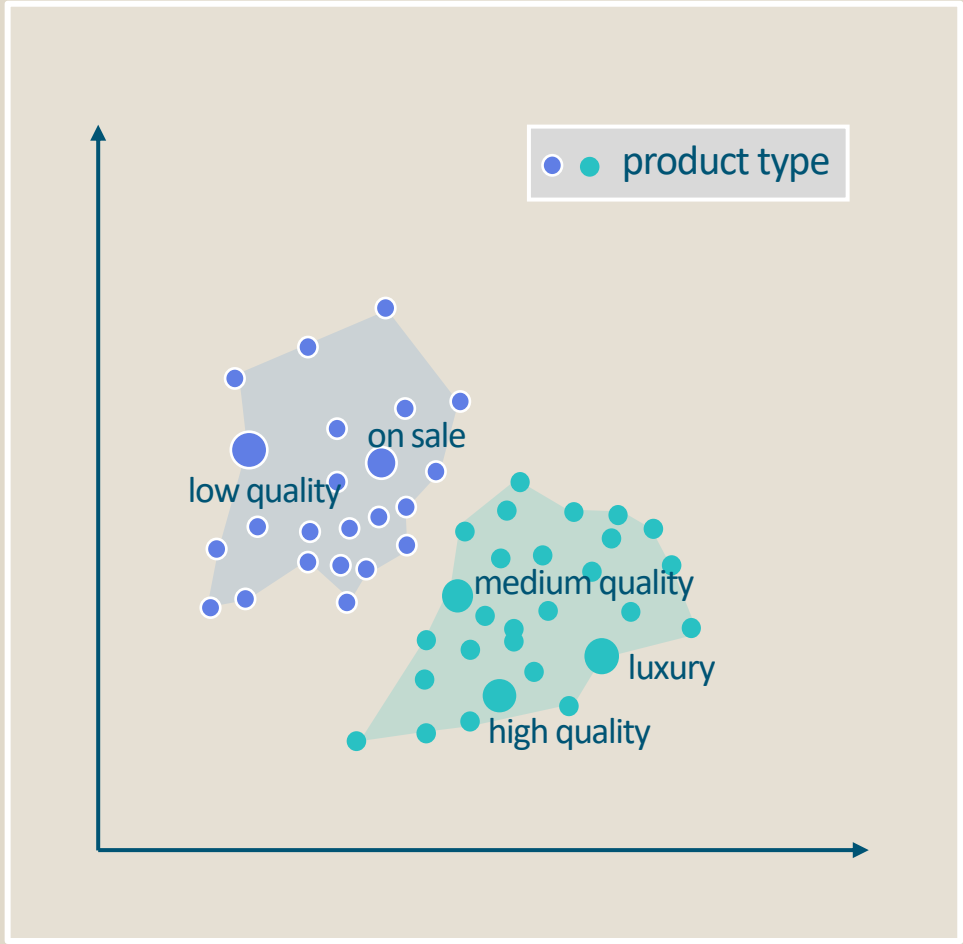
Grouping a set of data in such a way that samples within the same group are more similar to each other than to samples in different groups. The groups are called **clusters.**

Example

Your boss wants you to find out what preferences your customers have when buying in the company‘s online shop. To analyze the customer behavior a cluster analysis might help.

Input

Table of customers including the information of the bought items and their (categorical) attributes (type, price, color, size, sale (yes/no), quality, etc)



K-Means

Algorithm

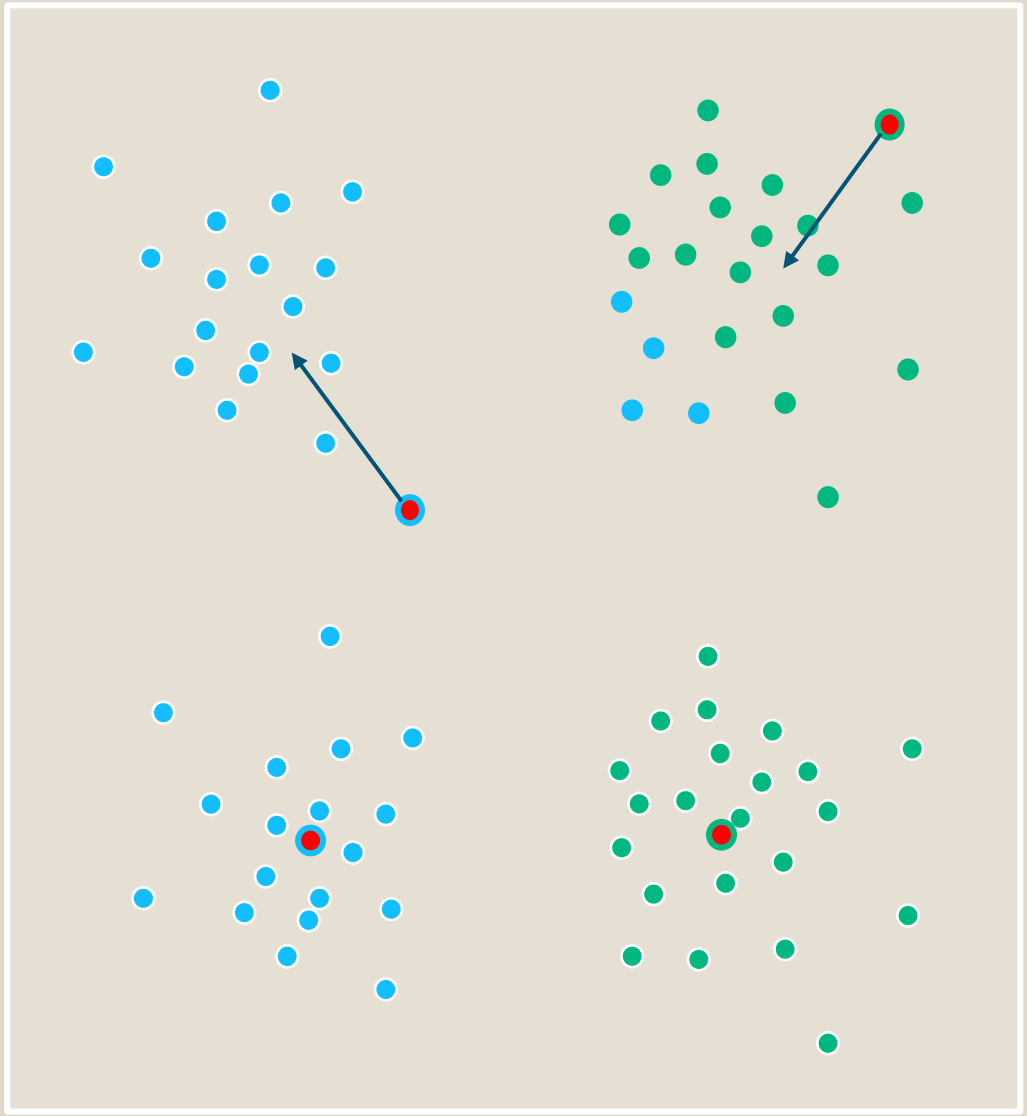
- Start with two randomly initialized centroids c
- Repeat those steps until convergence
  - Assign each point x to the closest centroid
  - Move the centroids to the center of the cluster
  - Reassign the points

Pros

- Works for many different problems and domains
- Scales well for large data

Cons

- Suffers from curse of dimensionality
- Performs poorly on elongated clusters
- Requires the number of clusters
- Has no concept of outliers



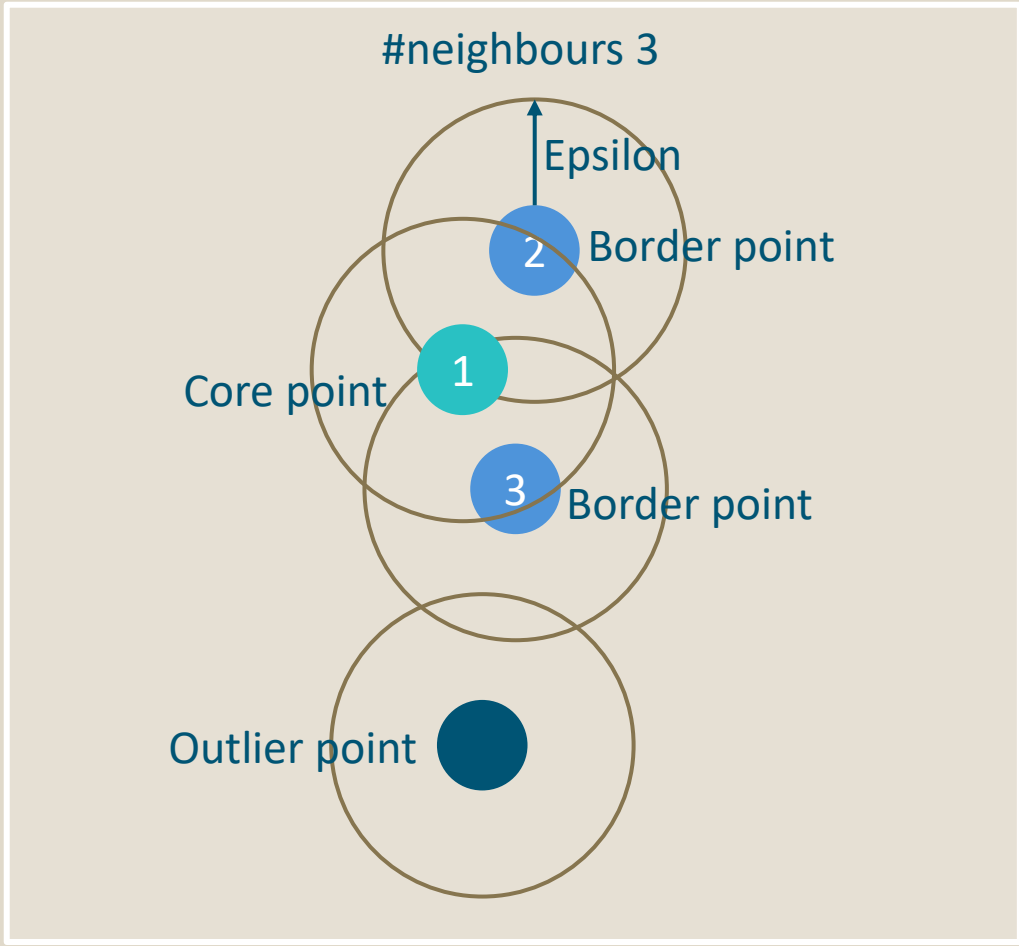
Density clustering with DBSCAN

DBSCAN: Density-Based Spatial Clustering of Applications with Noise

DBSCAN sees clusters as areas of high and low density

Algorithm

- Calculate the distance from each point to each other point in the dataset
- All points that have more than #neighbours points in a radius of epsilon will be classified as core points
- All points that have less neighbours but are a neighbour of a core point are called border points
- All points that have neither are classified as outliers
- Connect all core points to their neighbouring core and border points
- All directly connected points are called a cluster



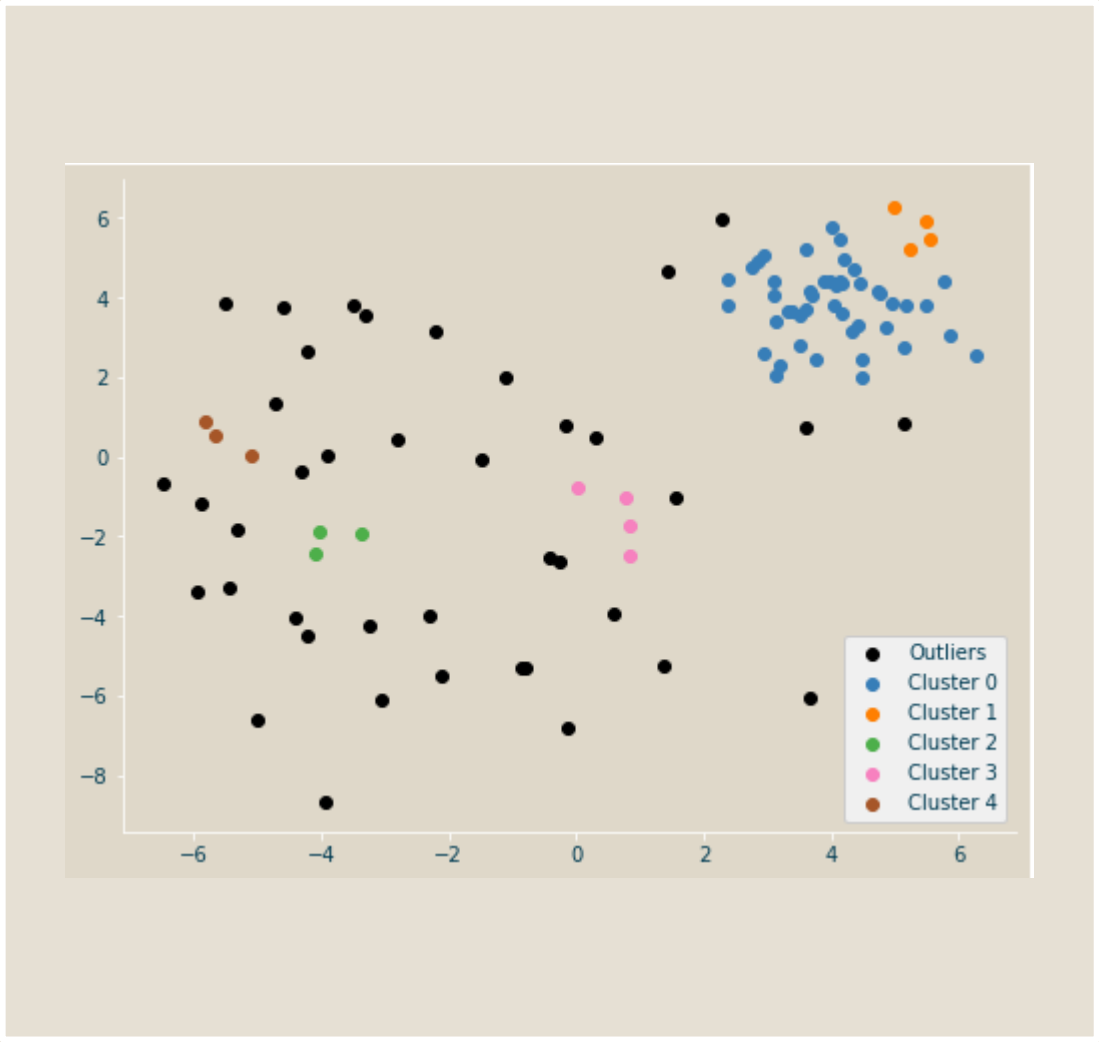
Density clustering with DBSCAN

Pros

- Can find clusters of any shape
- No need to specify number of clusters

Cons

- DBSCAN has problems when the clusters have different densities
- Need to tune the parameters #neighbours and radius epsilon



Hierarchical Clustering

Algorithm

- Start: each data point in on cluster $C_i = \{x_i\}$
- Repeat until one cluster left:
  - Find two clusters that are close, i.e., $\min_{i,j} D(c_i, c_j)$ for D defined with help of d, e.g.,
    - Single linkage: $D(c_i, c_j) = \mathbf{min}\{d(x, y) \text{ for } x \in c_i, y \in c_j\}$
  - Merge closest clusters c_i, c_j to c_{i+j}

Alternative

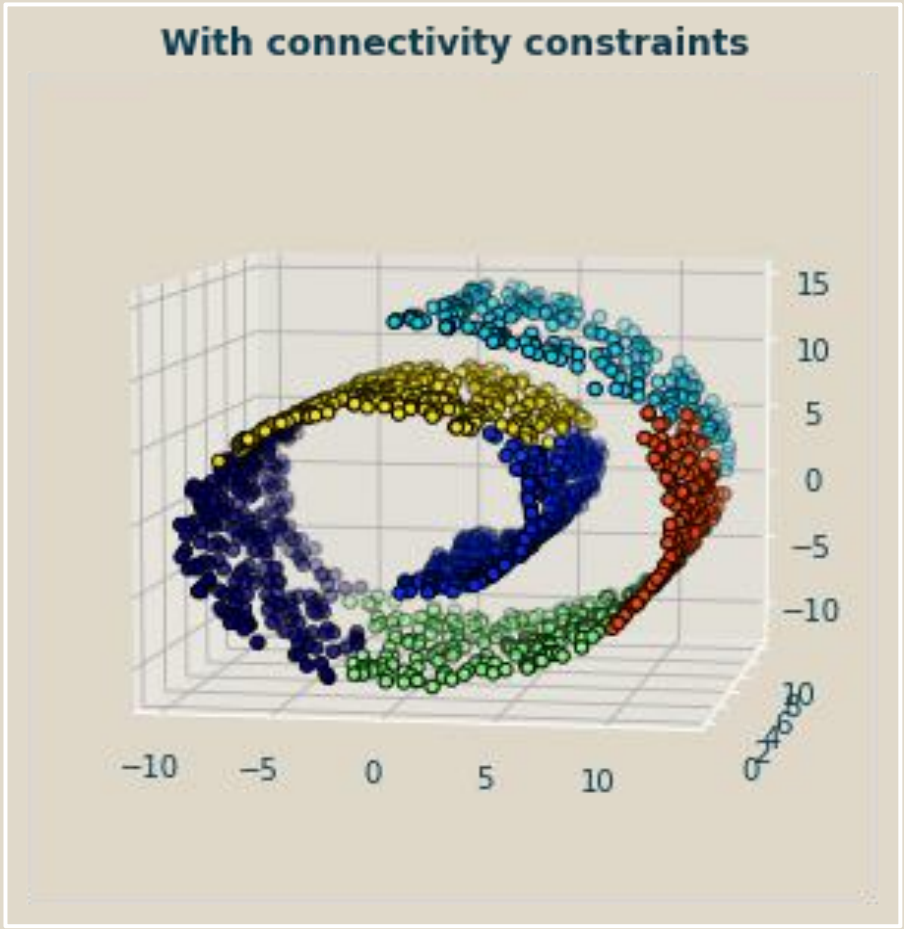
- Complete linkage: $D(c_i, c_j) = \mathbf{max}\{d(x, y) \text{ for } x \in c_i, y \in c_j\}$
- Average linkage: $D(c_i, c_j) = \mathbf{mean}\{d(x, y) \text{ for } x \in c_i, y \in c_j\}$

Pros

- Allows adding connectivity constraints, to only merge directly adjacent clusters
- Connectivity constraints greatly increase the scalability of the algorithm

Cons

- Can be computationally expensive



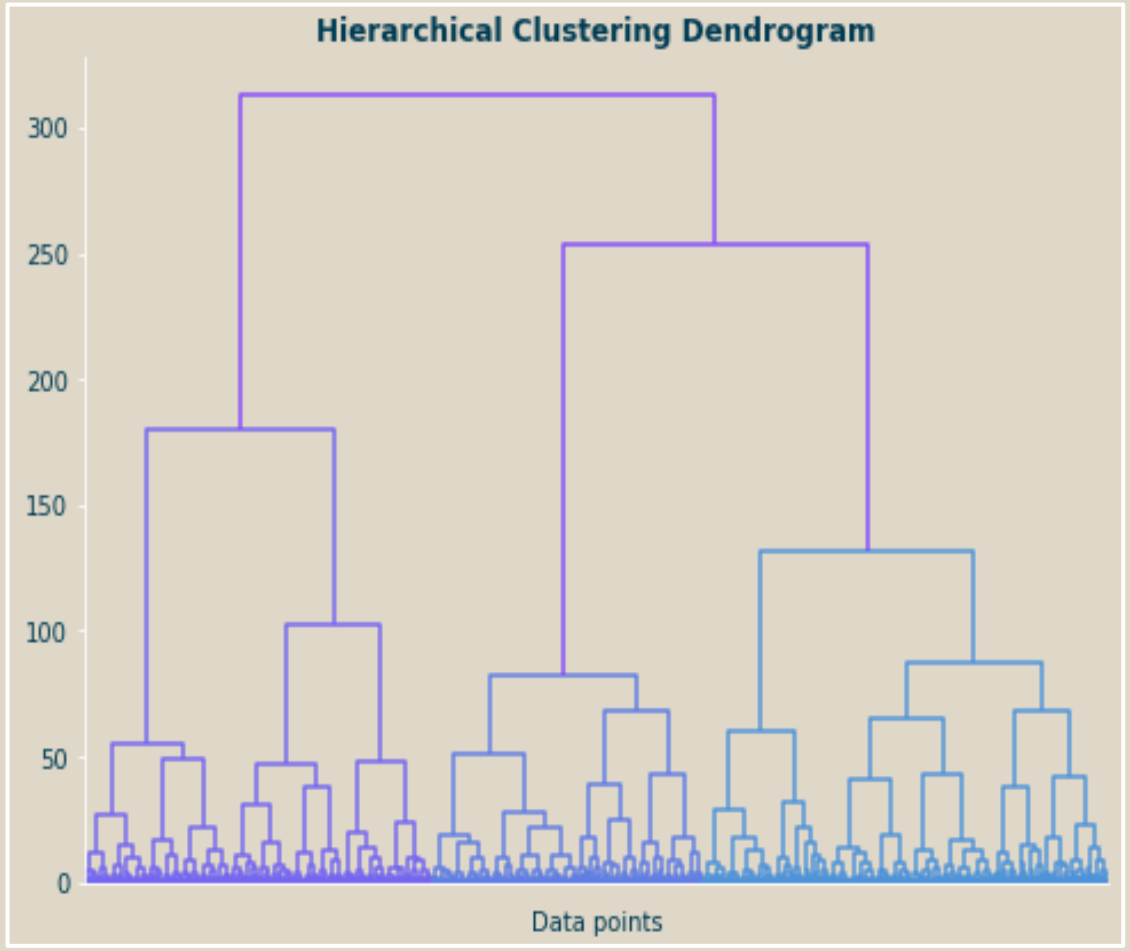
Connectivity constraints allow to only merge directly adjacent clusters

Hierarchical Clustering

Dendrogram

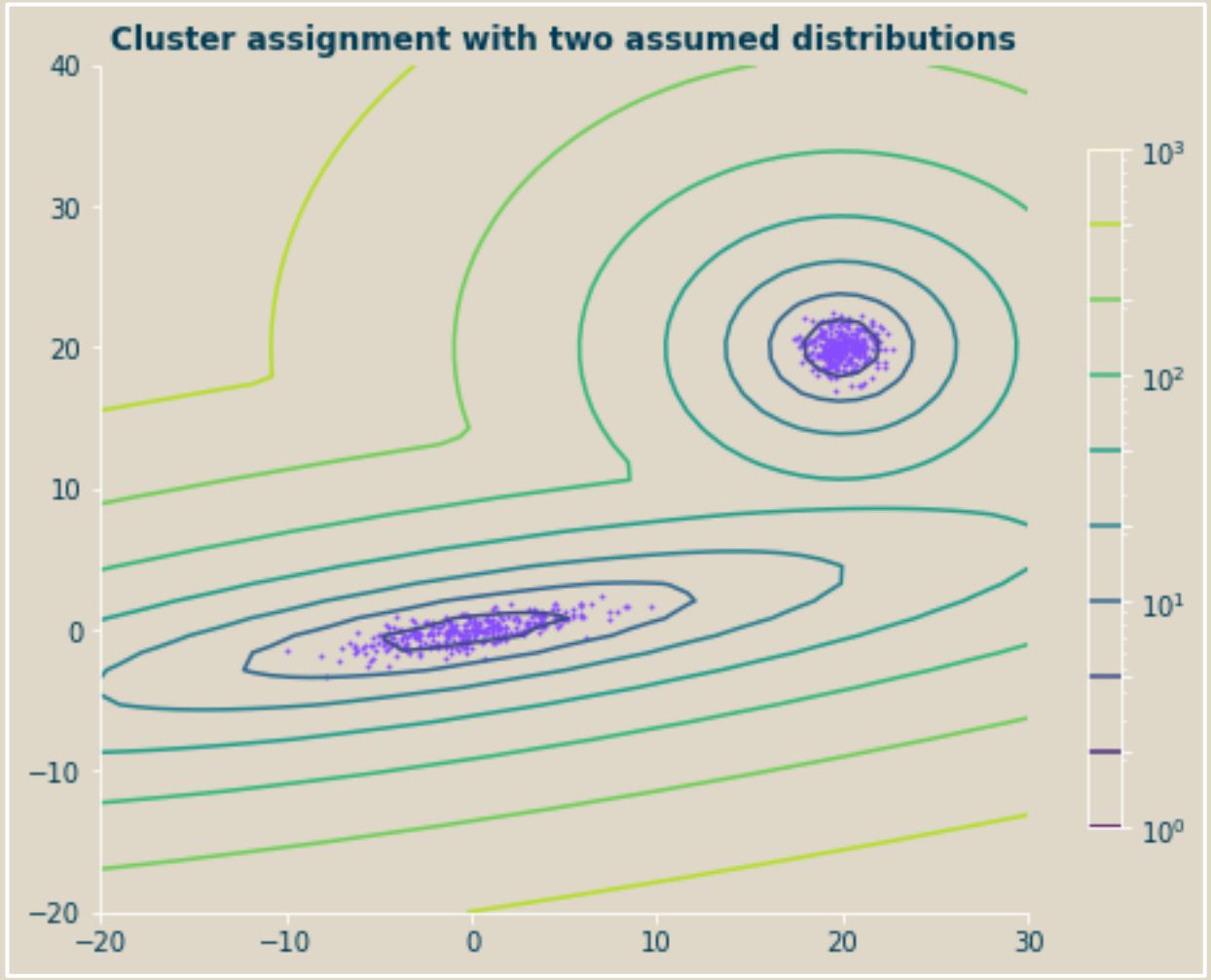
- Visualize the merging process of samples in a tree structure
- Decide the number of clusters after the clustering ran
- The cutoff point will determine the number of clusters that weren't merged yet

Dendograms are best used for **small sample sizes**



Gaussian Mixtures

- Distribution-based clustering with predefined number of clusters k
- Assumes that the data points are generated from a mixture of Gaussian distributions with unknow means and variances
  - Think of it as each cluster being distributed according to a Gaussian distribution
- Clustering means finding the best parameters to explain the data



Soft clustering with gaussian mixtures

Algorithm

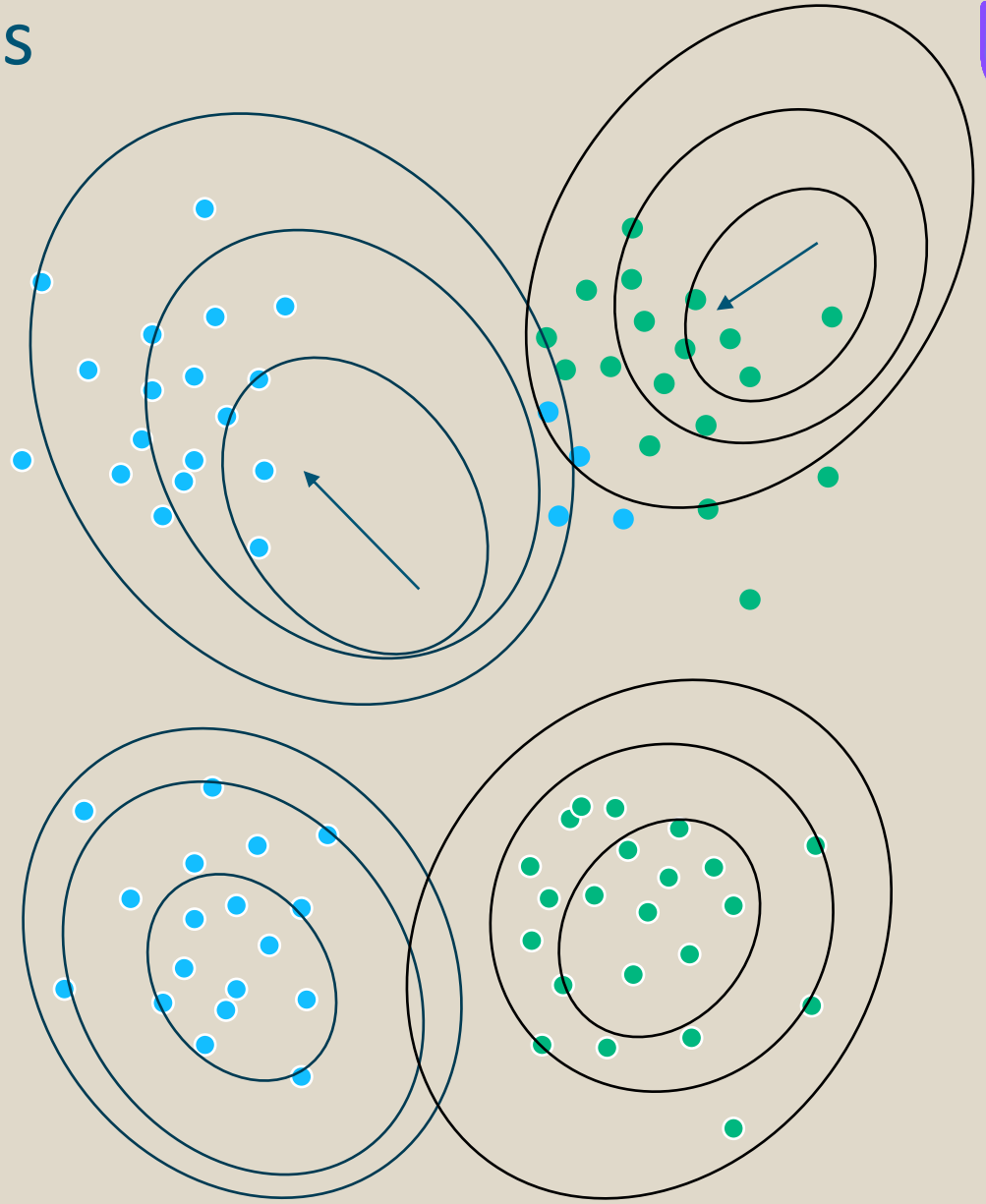
- Start with random or clever chosen means and variances
- For each point compute the likelihood that it comes from each distribution in the mixture
- Update the means and variances based on the likelihood of all data points

Pros

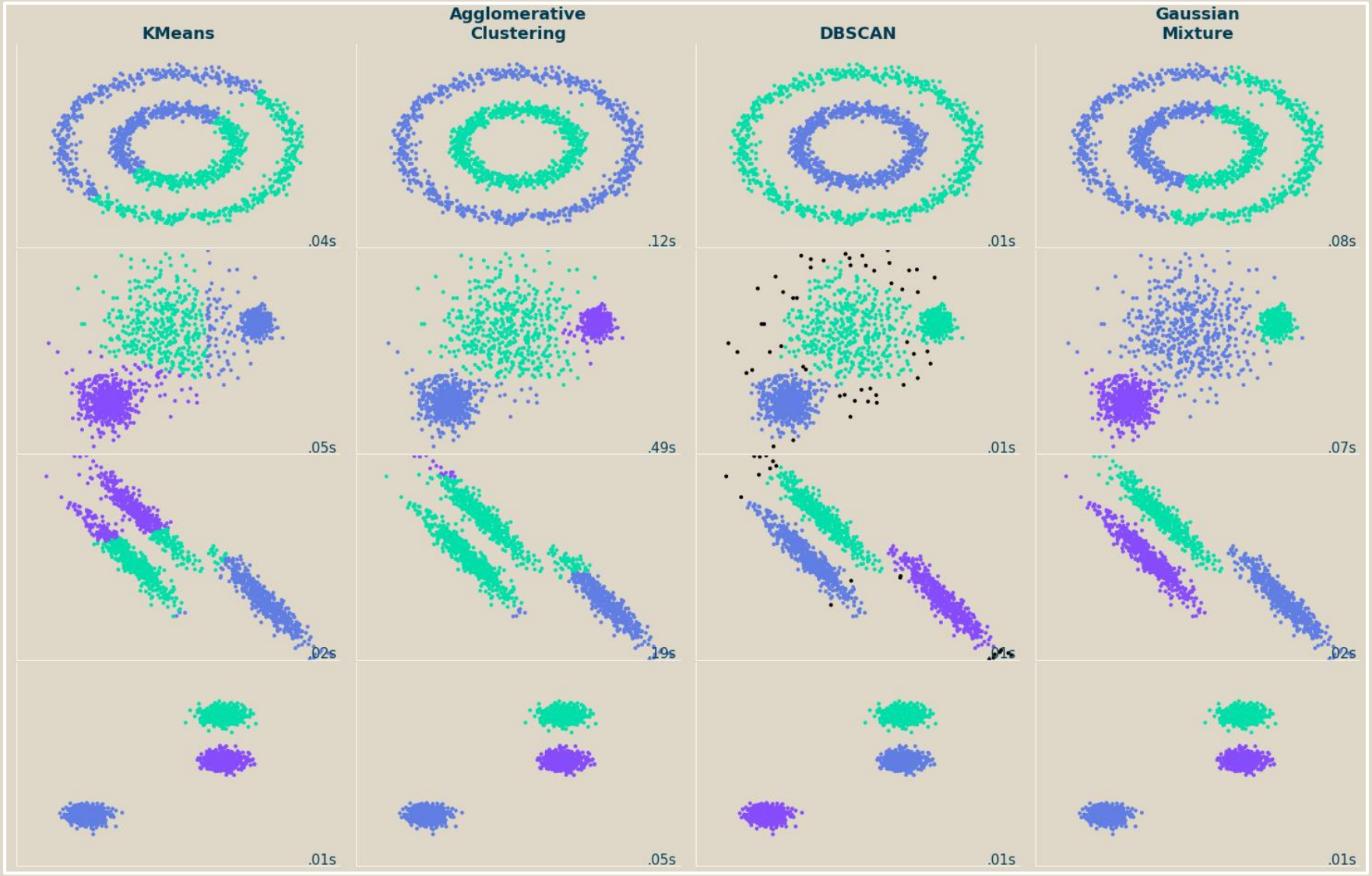
- Detects circles and ellipsoids
- Soft clusters: For each point we get the probability that it belongs to a given cluster

Cons

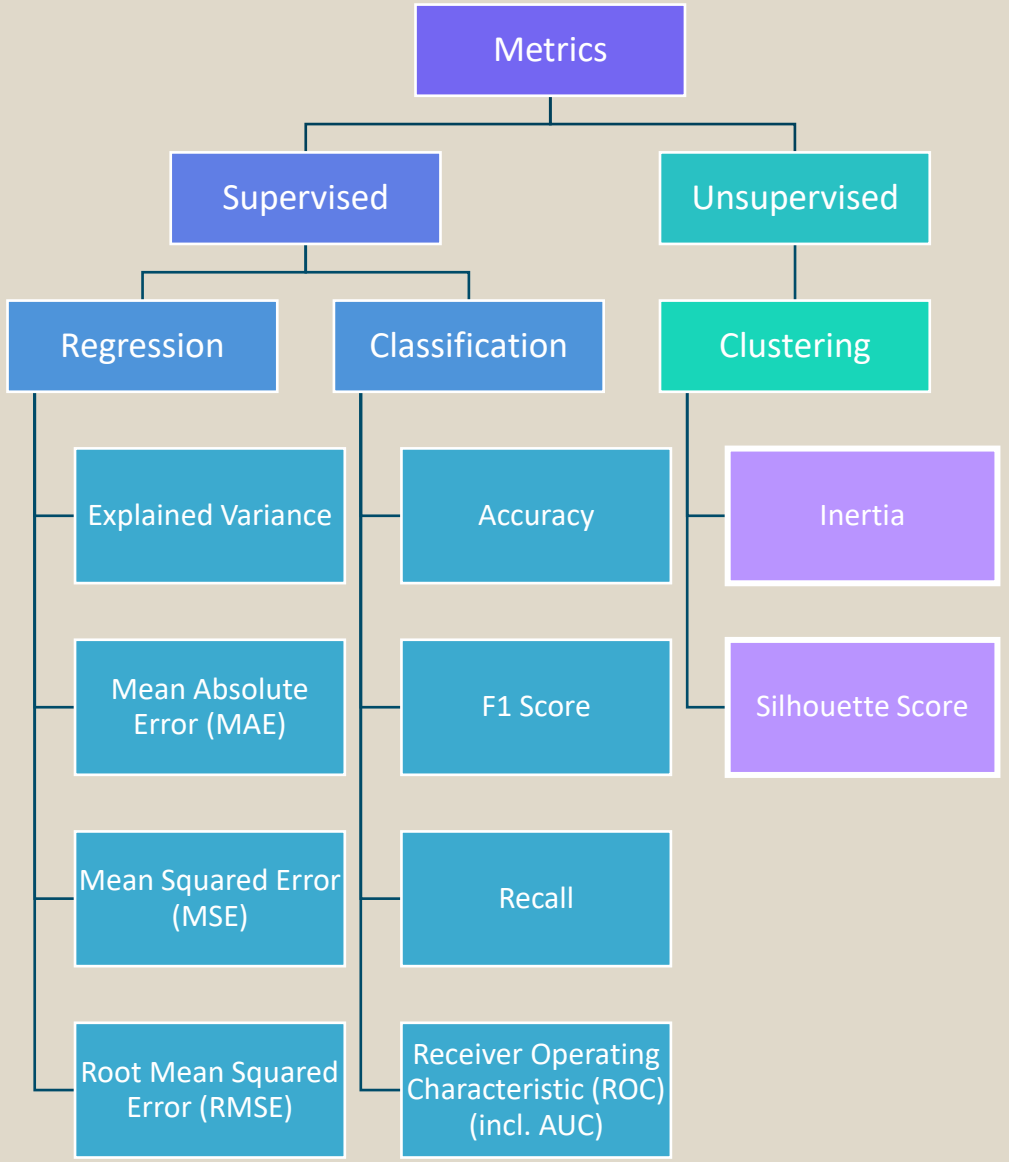
- Number of clusters must be fixed in advance
- Unstable if number of samples is too low
- Slower than k-means



Overview Clustering Algorithms



Selected evaluation metrics



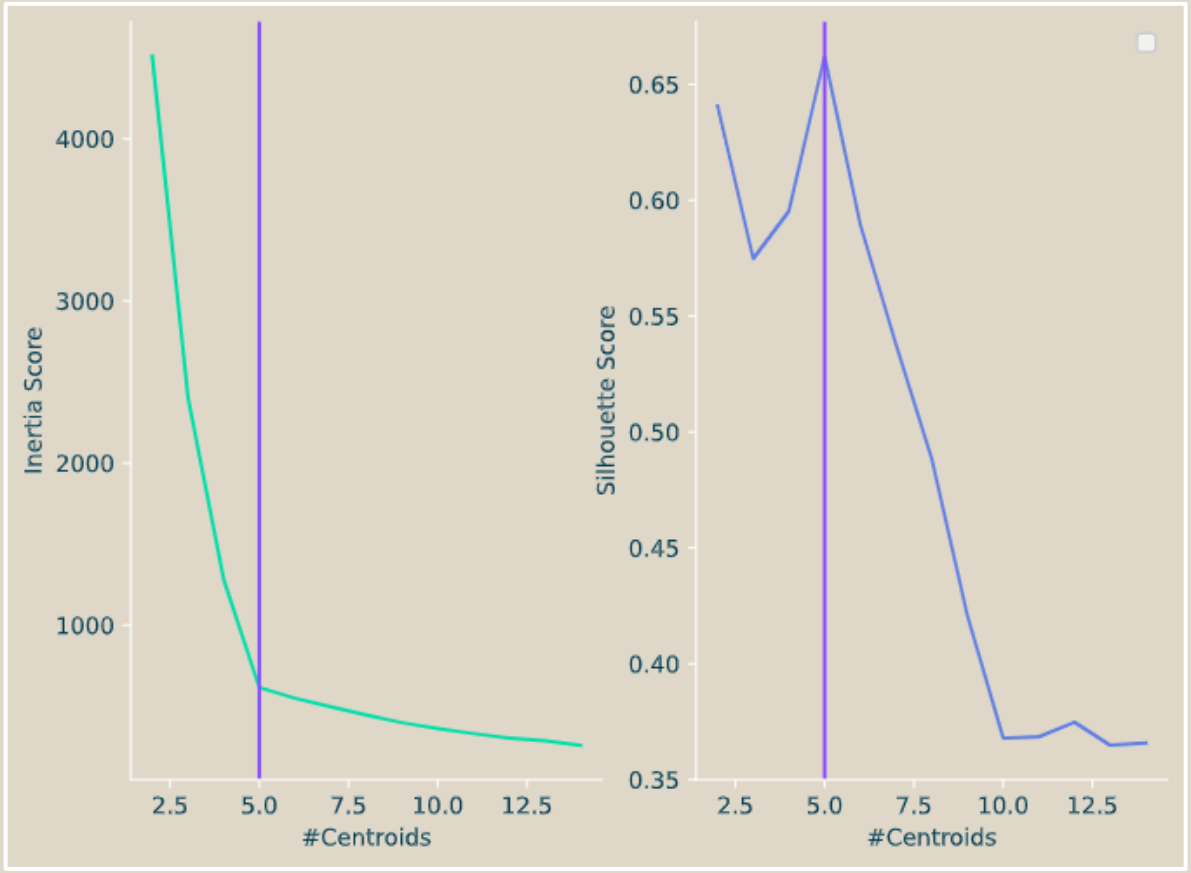
Evaluation metric for clustering

Within-cluster sum-of-squares criterion (inertia)

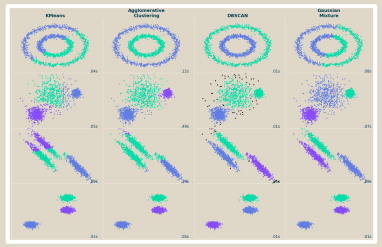
- Measures the **sum of squared distances** between each datapoint and its assigned cluster center
- **Increasing the number of clusters decreases** the within-cluster sum-of-squares criterion

Silhouette Score

- Calculates the **difference between the average distance** of samples **within a cluster** and the average distance between a sample and the **nearest neighboring cluster**
- A value of **1 is the best possible value**. A value of **0** indicates **overlapping clusters**. A value **-1** means points were assigned to the **wrong cluster**

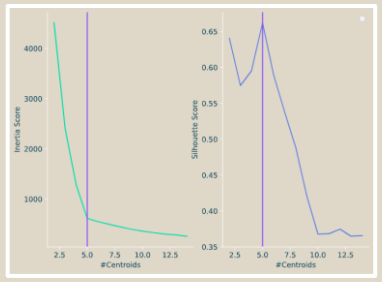


Important takeaways



Depending on our problem and data different **clustering algorithms** can summarize and describe our data

- **KMeans** partitioned the data by moving around centroids
- **Agglomerative** clustering builds hierarchical clusters by merging the nearest two clusters
- **DBSCAN** connects densely connected regions of samples
- **Gaussian Mixtures** use distributions of centroids for soft-clustering



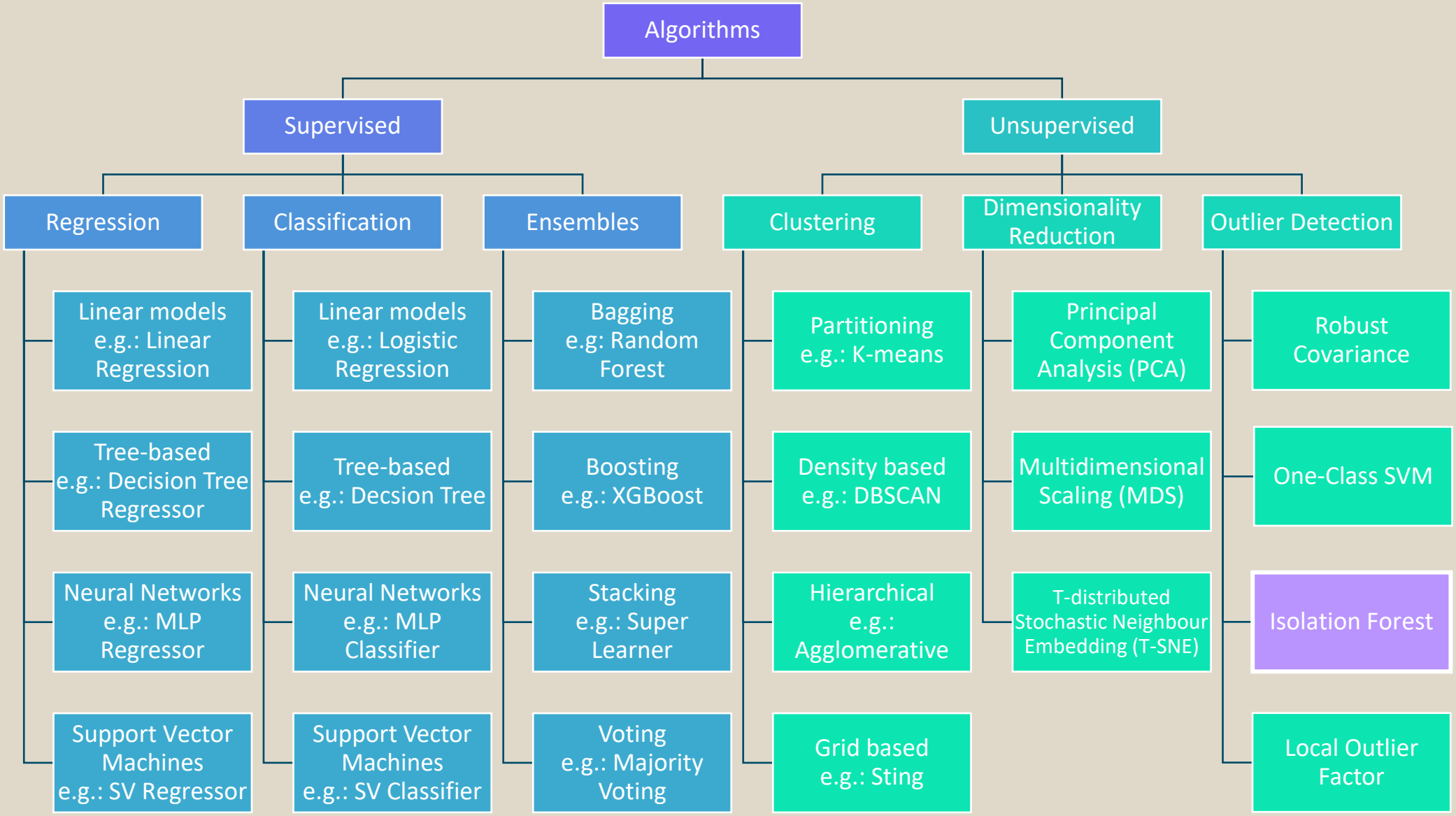
For some clustering algorithms we need to **specify the number of clusters**. We can do that by using cluster evaluation metrics:

- Use the **inertia** score from the KMeans algorithm with the **elbow method**
- Choose the number of clusters that maximizes the **silhouette score**

Try it yourself!

In the following exercises

Selected model types



Outliers

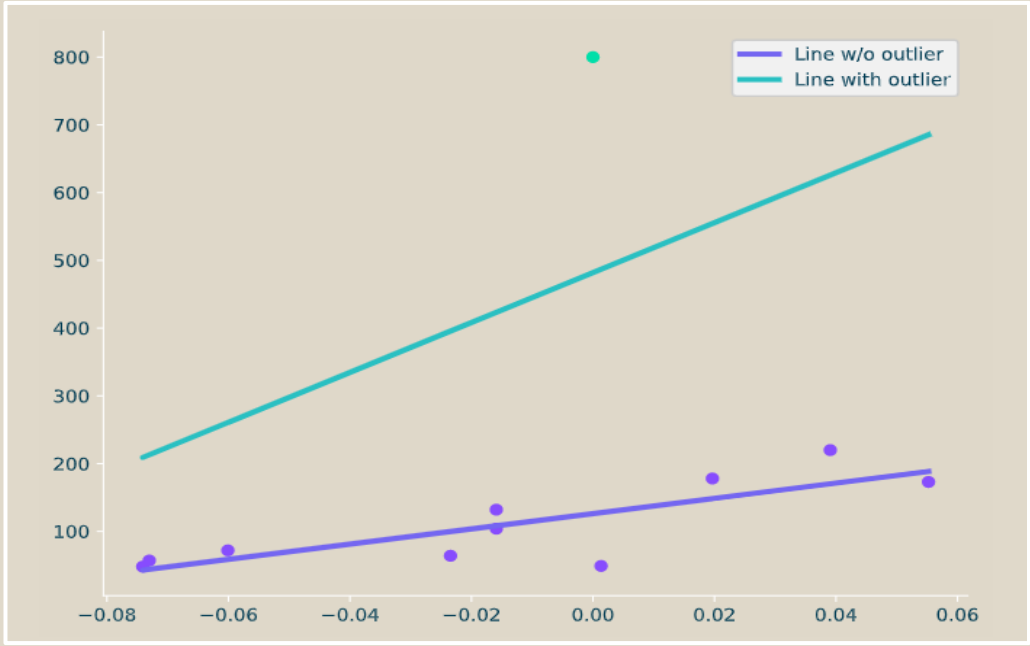
Outliers are observations that diverge from an overall pattern on a sample

Causes of outliers:

- Measurement or input error
- True outlier observation

Types of outliers

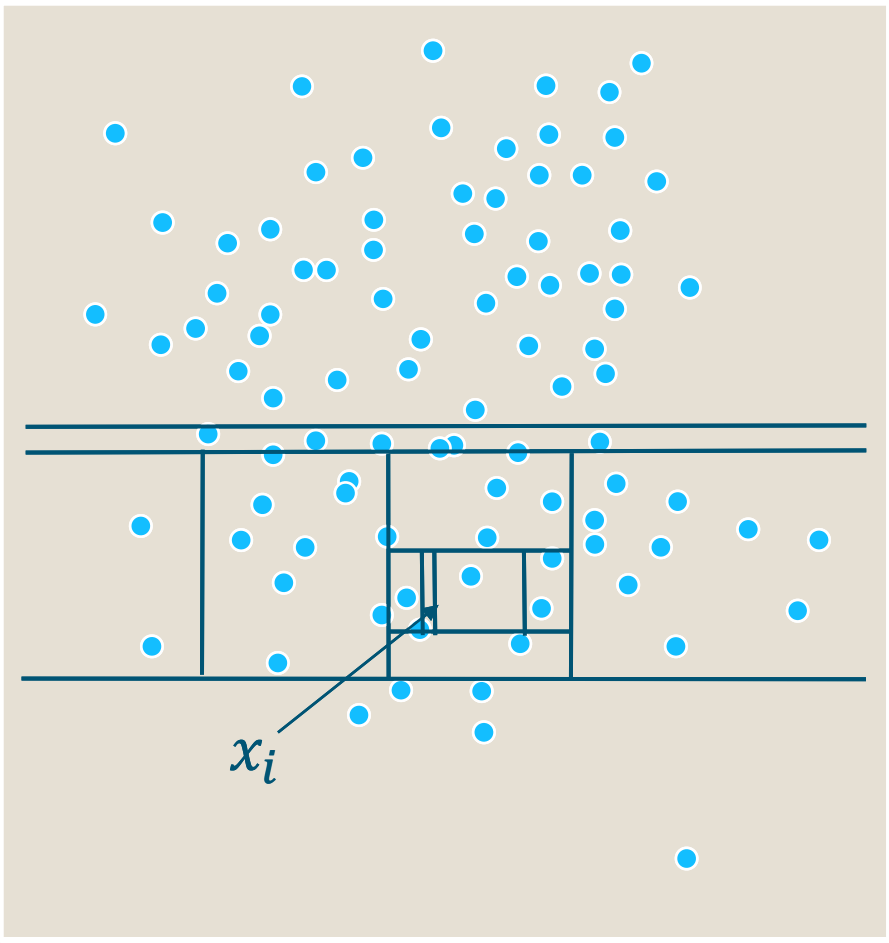
- Univariate: Found when looking at the distribution of values of a single feature
  - Univariate unrealistic values can just be filtered
- Multivariate: Is found when looking at the outlier in n-dimensional space
  - Collective and multivariate outliers we need a model to encode what is normal

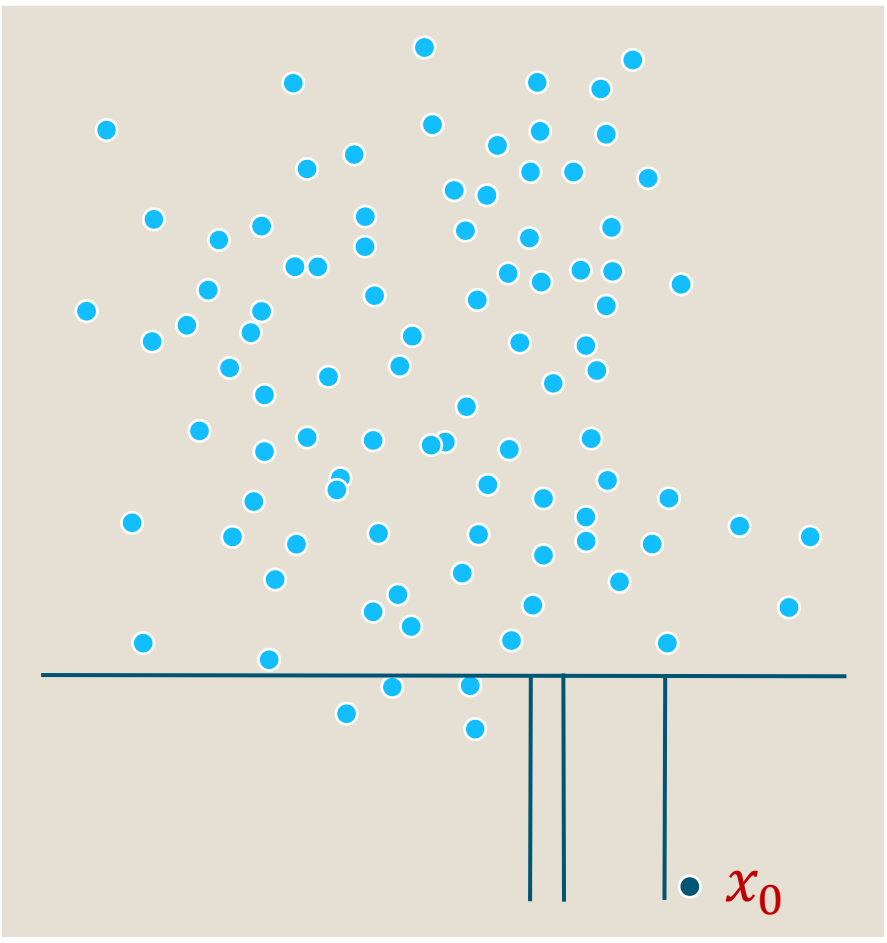


Isolation Forest: Random Tree

Create a random tree as follows:

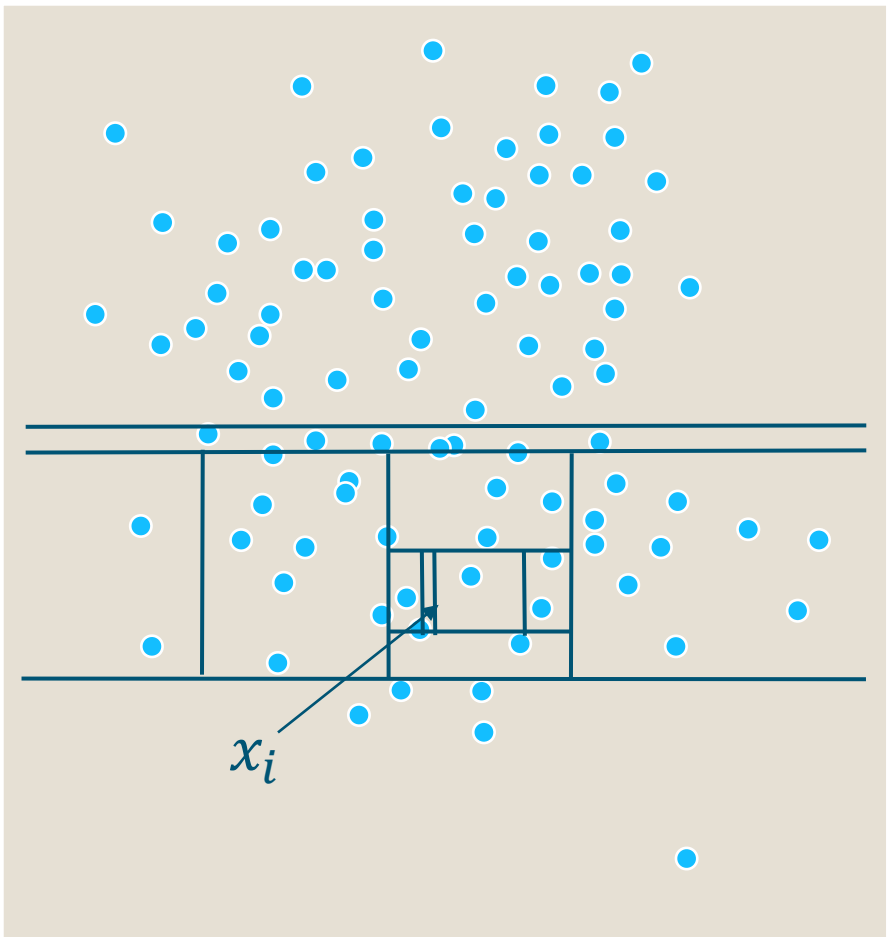
- Repeat until single sample in node or set depth reached:
  - Create split by randomly select a feature and a random split value between minimum and maximum of that feature

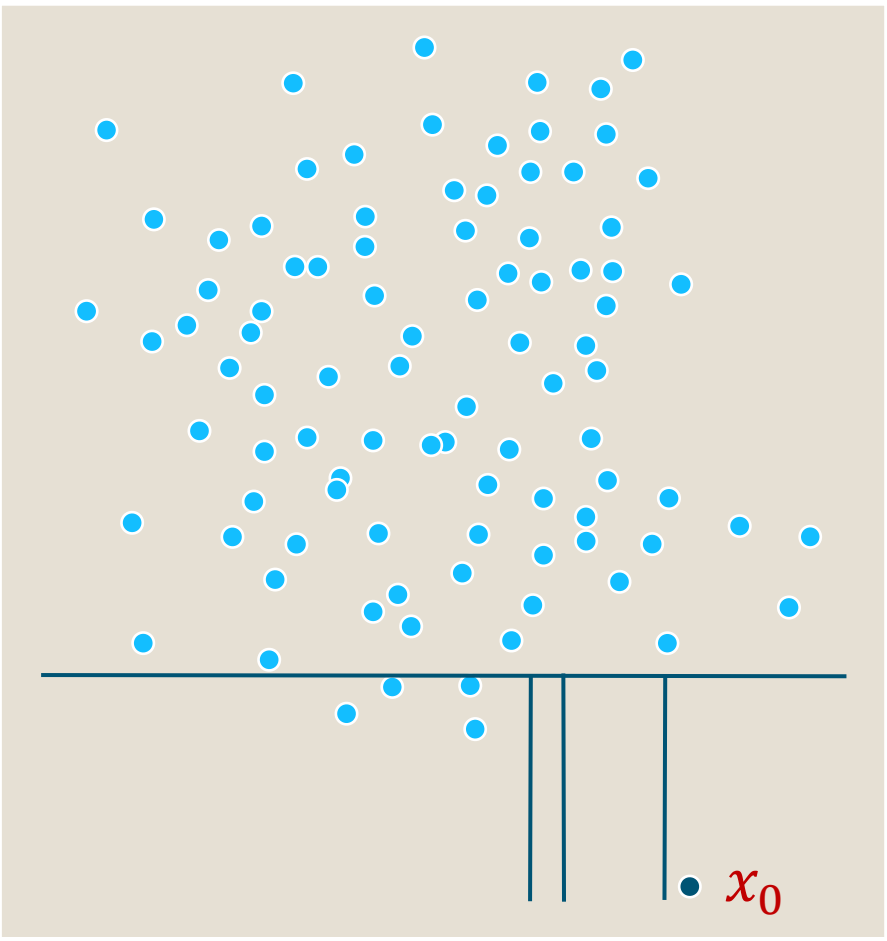




Isolation Forest: Outlier detection

- Create a lot of the random trees from previous slide
- For each sample: Compute a score based on the average path length (i.e., number of splits in the tree until we end up a that sample)
  - Outlier x_0 will have a lower score since the average path length smaller (x_0 is easier to "isolate")





Outlier Detection: Isolation Forest

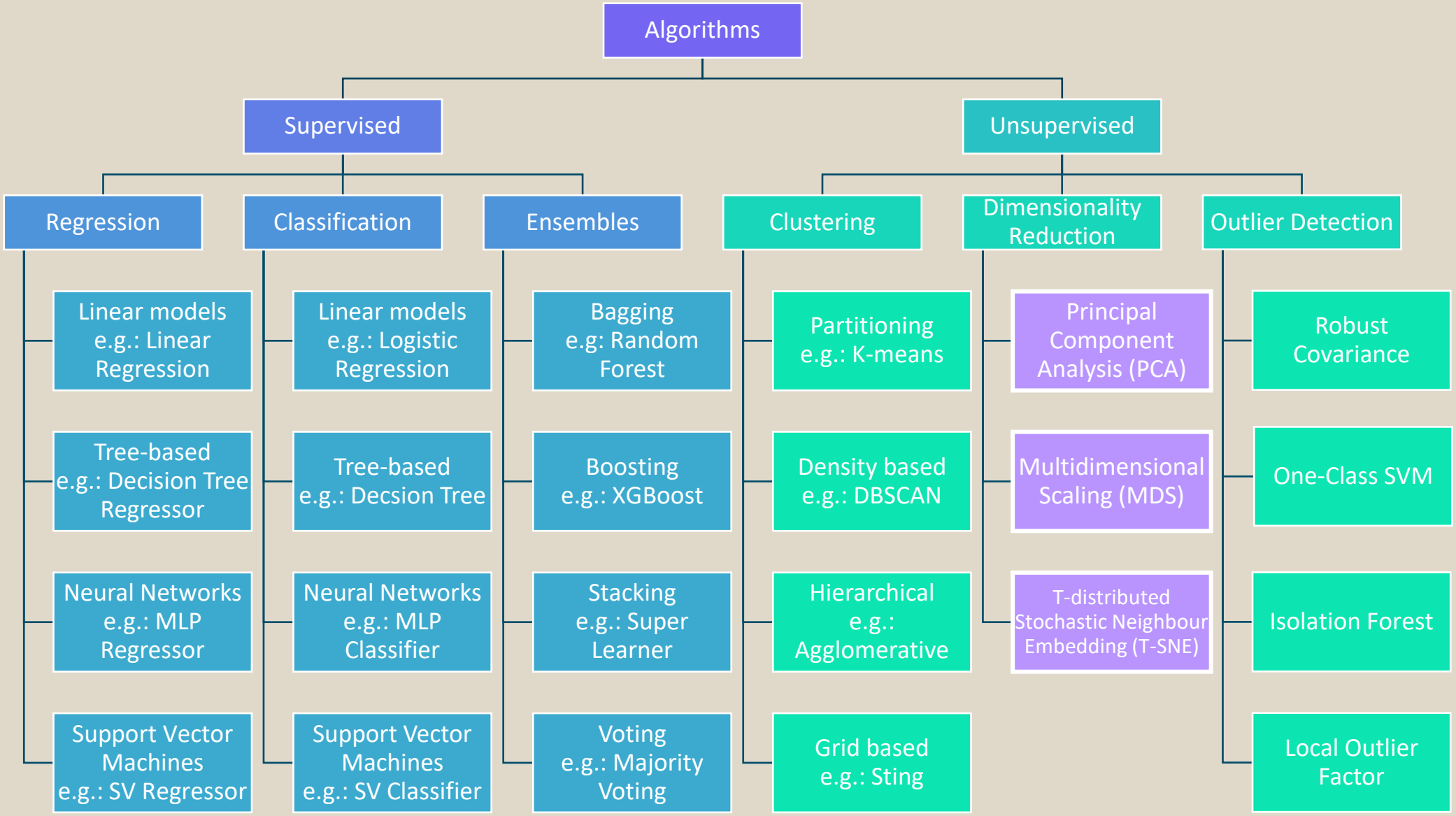
Pros

- Non-parametric
- No scaling of features necessary
- Trainable without anomalies in the dataset
- Can handle high dimensionality and irrelevant features
- Scales well in terms of computation time and memory for large data

Cons

- Cuts are always parallel to feature axis, sometimes cause problems
- Large numbers of anomaly points in the training data can reduce the detection capabilities

Selected model types



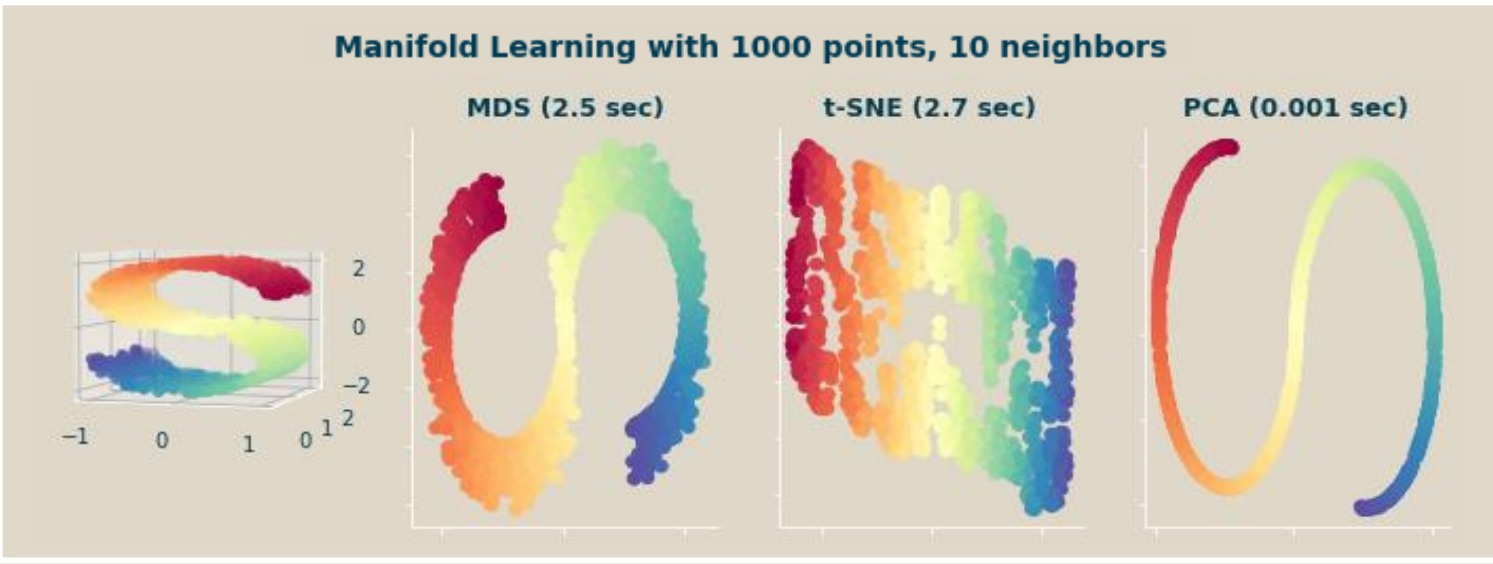
Dimensionality Reduction

Goal

- **Reducing the number of features**, i.e. the dimensionality of the data set
- Feature selection: only use a subset of the available features
- Feature extraction/projection: Transform data from high-dimensional space to a space of fewer dimensions

Why to do it?

- High-dimensional data sets are hard to understand for humans and some models (this is sometimes called the curse of dimensionality)
- Speed up model training and evaluation
- Reduces noise and redundant/linear correlated feature in data (especially PCA) might increase model performance



Dimensionality Reduction: PCA

- Principal component analysis computes new features that are linear combination of the given features: given features f_1,…, f_d compute new features

$$\tilde{f} = a_1 f_1 + \ \ldots + a_d f_d$$

- New features $\tilde{f_1}, \ldots, \tilde{f_d}$ are computes in such a way that $f_1\tilde{}$explains more variance of the data than$f_2\tilde{}$ that explains more variance than $f_3\tilde{}$ and so on, i.e., the new feature are sorted by importance
- Dimension reduction by using only the first k new features

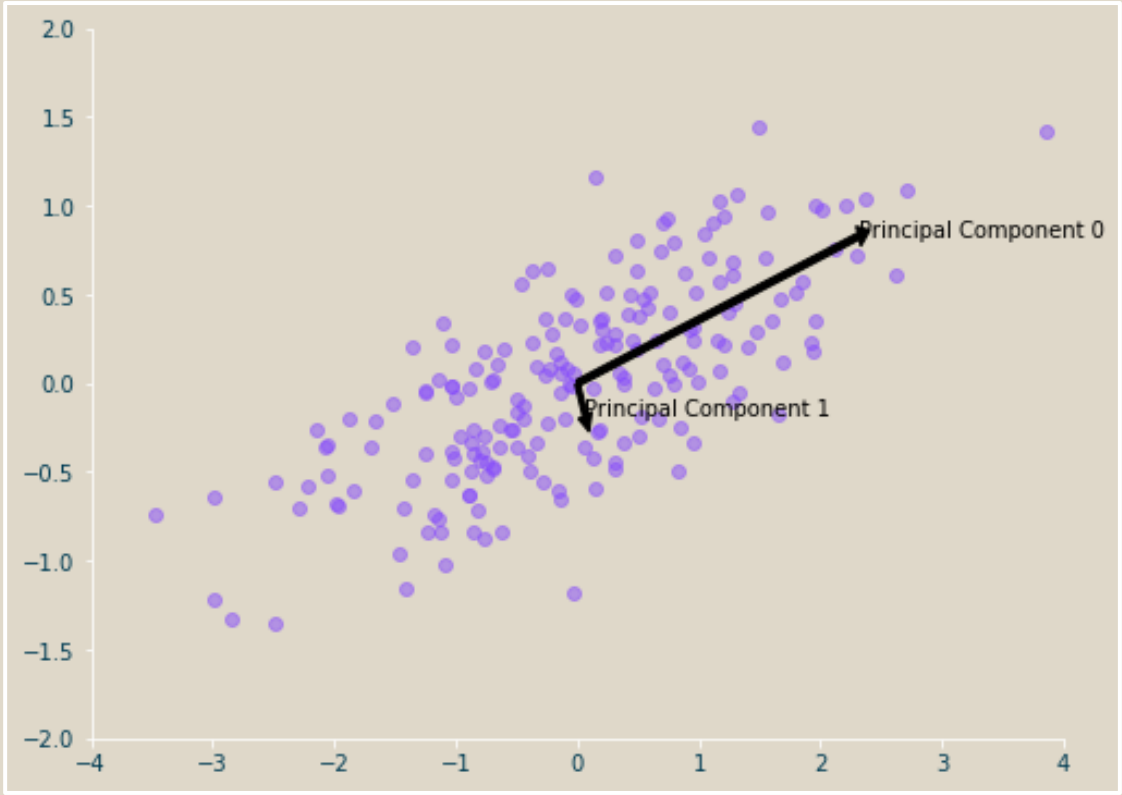
Pros

- Reduces noise and “removes” redundant (linear correlated) features

Cons

- New features might not have an understandable interpretation
- Can only perform linear transformations

Note: PCA requires normalization of each feature separately if the features have different scales. Otherwise the features with the biggest values will always be more important



Dimensionality Reduction: MDS

- MDS: (metric) Multi-dimensional scaling
- Given high-dimensional or complex data $x_1, \dots x_n$ and a distance function d, compute all pairwise distances $d_{i,j} = d(x_i, x_j)$
- Find representatives $\bar{x}_1, \dots, \bar{x}_n \in \mathbb{R}^2$ that minimize

$$\sum_{i \neq j = 1,\dots,n} \left(d_{i,j} - \|\bar{x}_i - \bar{x}_j\|\right)^2$$

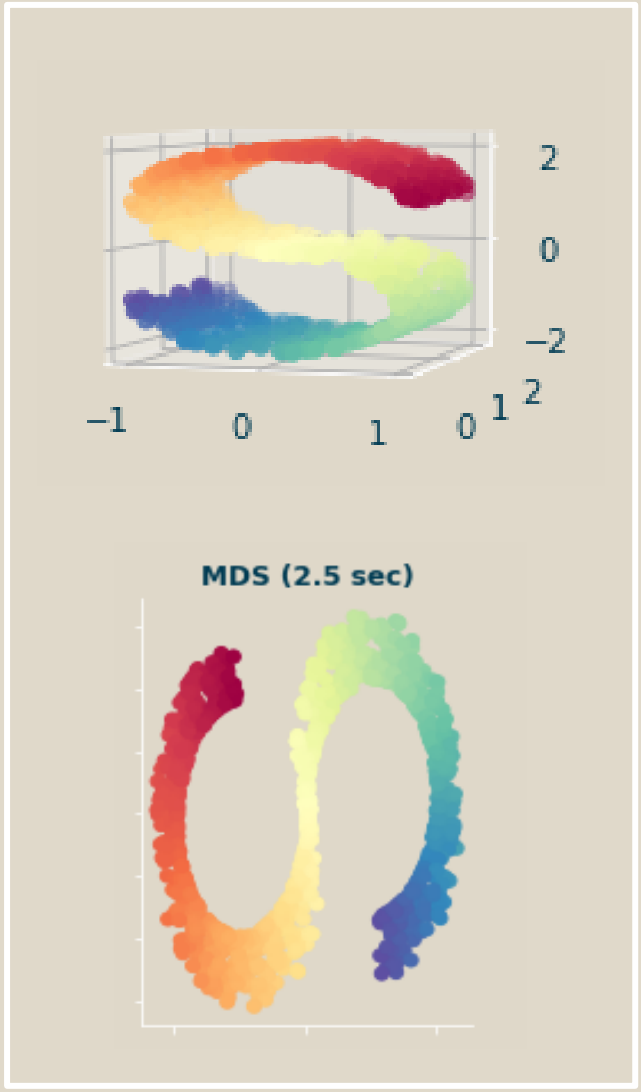
- Creates basically a 2d map of the samples
- E.g., samples are chemical compounds and MDS is used to create a map where similar compounds end up close together

Pros

- Nice visualizations that are easy to understand for experts if the distance makes sense

Cons

- Slow for even medium data sets
- Reduction is often too strong since distance to other samples is not enough information to train a model



t-distributed Stochastic Neighbor Embedding (T-SNE)

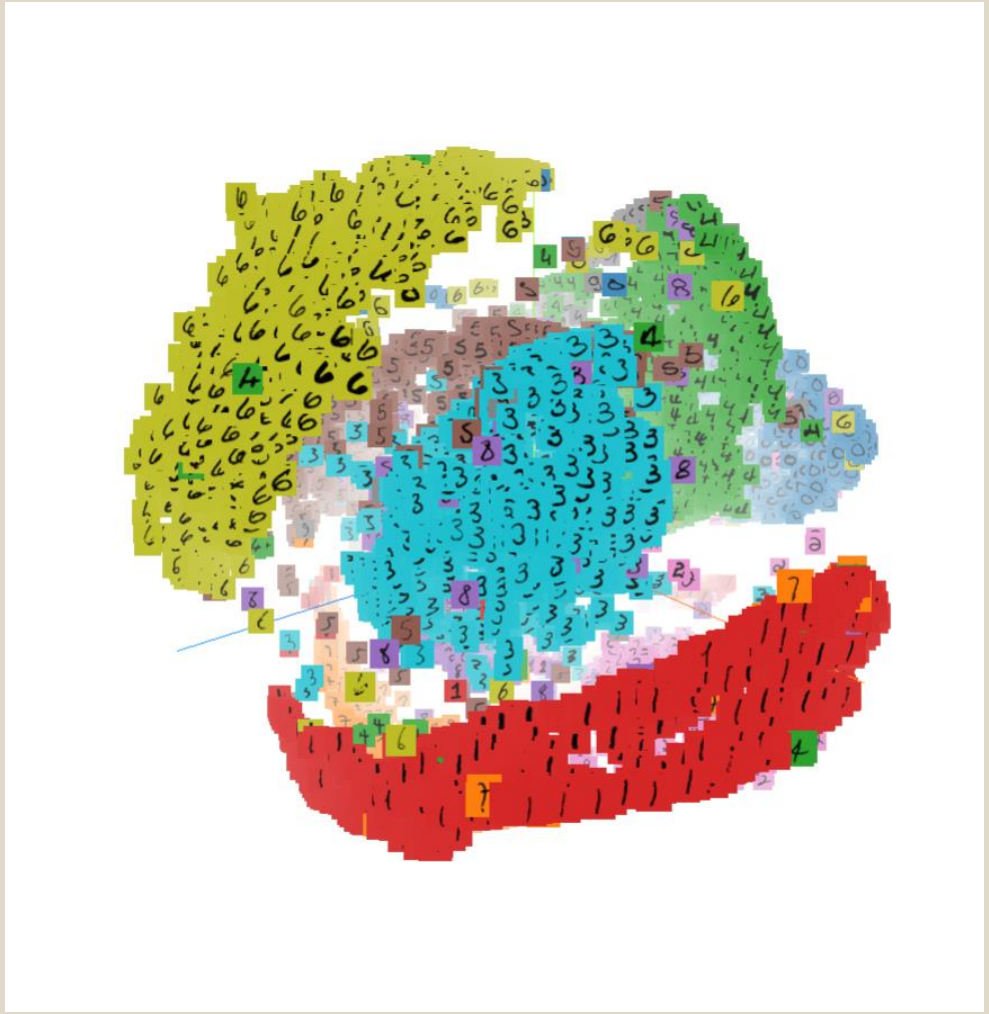
- Main purpose visualize high-dimensional data in two or three dimensions
- MDS reduces data into low dimensions while keeping the distances intact
- T-SNE reduces the dimensionality of the data while clustering points together that where also clustered in high dimensional space
- “Kullback-Leibler (KL) divergence of the joint probabilities in the original space and the embedded space will be minimized by gradient descent”

Advantages

- Reduce any data into low dimensional space
- Reveal relationships even if they exist in multiple sub-dimensions
- Avoids the tendency to crowd all data points into one center

Disadvantages

- Computationally expensive and takes a long time to finish compared to PCA
- The algorithm is stochastic and multiple restarts will lead to different outcomes
- The global structure is not explicitly preserved



Dimensionality reduction with Neural Networks

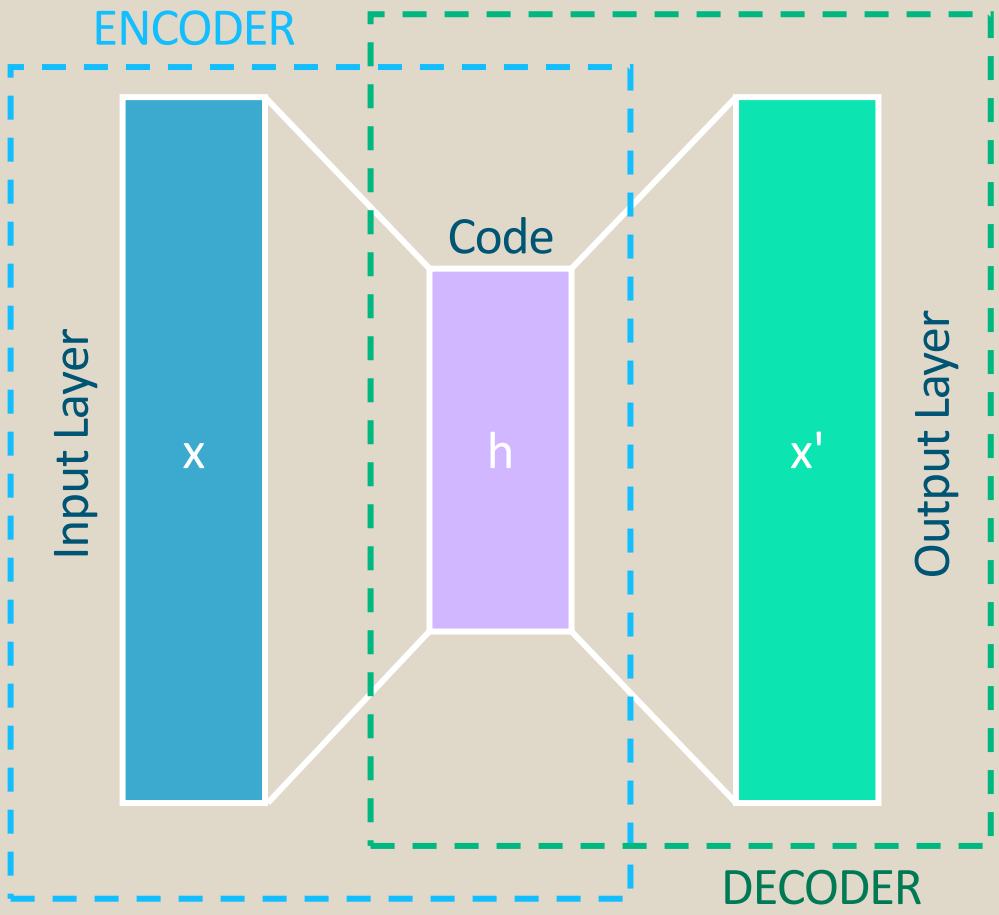
- Learn a neural network that takes an input and reduces the dimensionality into the layer h
- Then it reconstructs the input as x'
- The loss is the difference between x and x'

Autoencoders

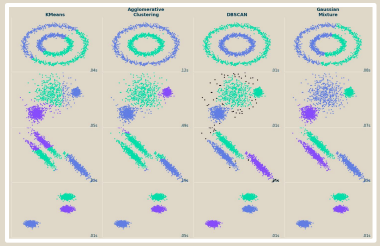
- Learn how to efficiently encode and com-press given data

Embeddings

- Very import in NLP (natural language processing), especially when working with texts
- Word2Vec embeddings: Learn a vector representation of words and their context

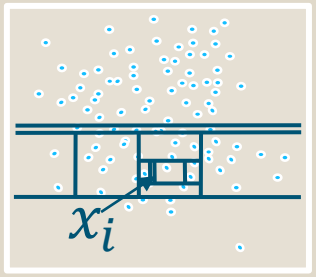


Important takeaways



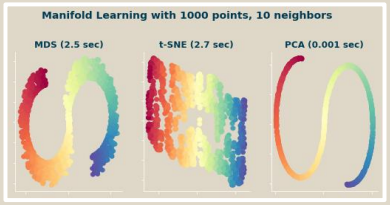
Depending on our problem and data different **clustering algorithms** can summarize and describe our data

- We have covered **kMeans**, **Agglomerative** clustering, **DBSCAN** and **Gaussian Mixtures**



Outliers are observations that diverge from an overall pattern on a sample

- **Univariate**: Found when looking at the distribution of values of a single feature
- **Multivariate**: Is found when looking at the outlier in n-dimensional space. We used the **Isolation Forest** to find these outliers.



With **dimensionality reduction** we reduce the number of features

- Principle Component Analysis (**PCA**) is often used to created a reduced and uncorrelated feature set
- Algorithms like **MDS** and **t-SNE** are used to represent high dimensional data in two or three dimensions

Outlier detection and dimensionality reduction is often used as a preprocessing step in the Data Science workflow

Quiz: Unsupervised Learning

Please join at slido.com with #031 077.

Let’s go through some questions together.

Let’s see what you think. All answers will be anonymous.

Try it yourself!

In the following exercises

Agenda

	Introduction
1	Recap Basic Machine Learning and Python
2	Complex Models
3	Model Evaluation
4	Hyperparameters
5	**Unsupervised Learning**
6	Gradient Descent
7	Deep Learning and Image Recognition
8	Deep Learning and Natural Language Processing
9	Repetition
10	Bias and Ethics in Machine Learning
11	Introduction to Data Science with AWS

Feedback and Q&A

Thank you

If you would like any further information please contact
Werner,Dr.,Fabian_Georg (BI X) BIX-DE-I
<fabian_georg.werner@boehringer-ingelheim.com>