Prof. Dr. Peer Küppers

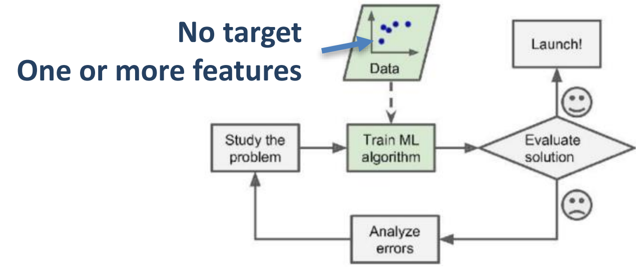
Summer School

Unsupervised Learning: Clustering

Agenda – Part 5

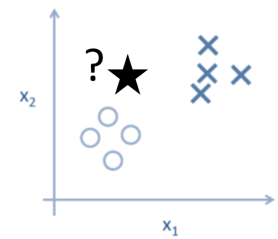
- **Introduction to Unsupervised Learning and Clustering**
- **Prototype-based Clustering (k-Means)**
- **Density-based Clustering (DBSCAN)**
- **Summary & Outlook**

Introduction

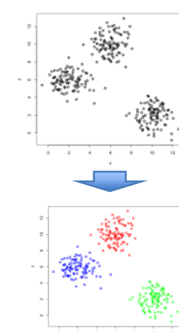


- **Clustering** is an unsupervised learning technique for identifying and grouping similar objects
  - labels attached to the clusters are not available in the historical data
  - we want to derive these labels.

- Supervised learning
  - Goal: predict target value (numerical=regression, categorical=classification) for samples with unknown target value
  - Search for and model dependencies between features and target

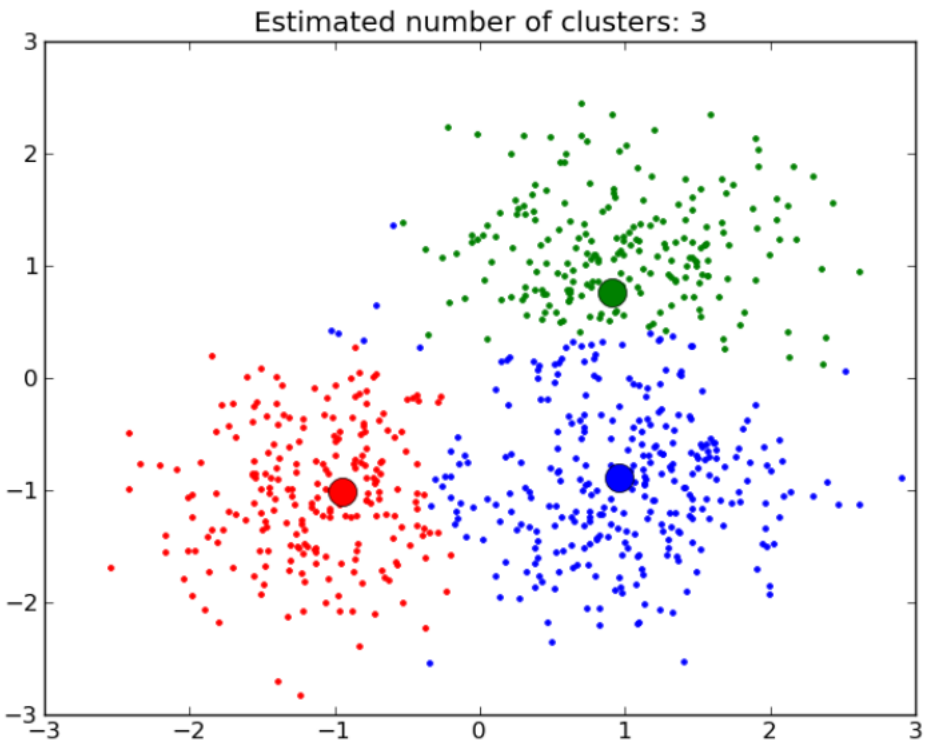


- Unsupervised learning
  - Goal: Create a pattern or a more compact description input data
  - No reference to target attribute, error not directly measureable.
    - We cannot hold back a fraction of the data (train-test-split) as in classification / regression!



Introduction

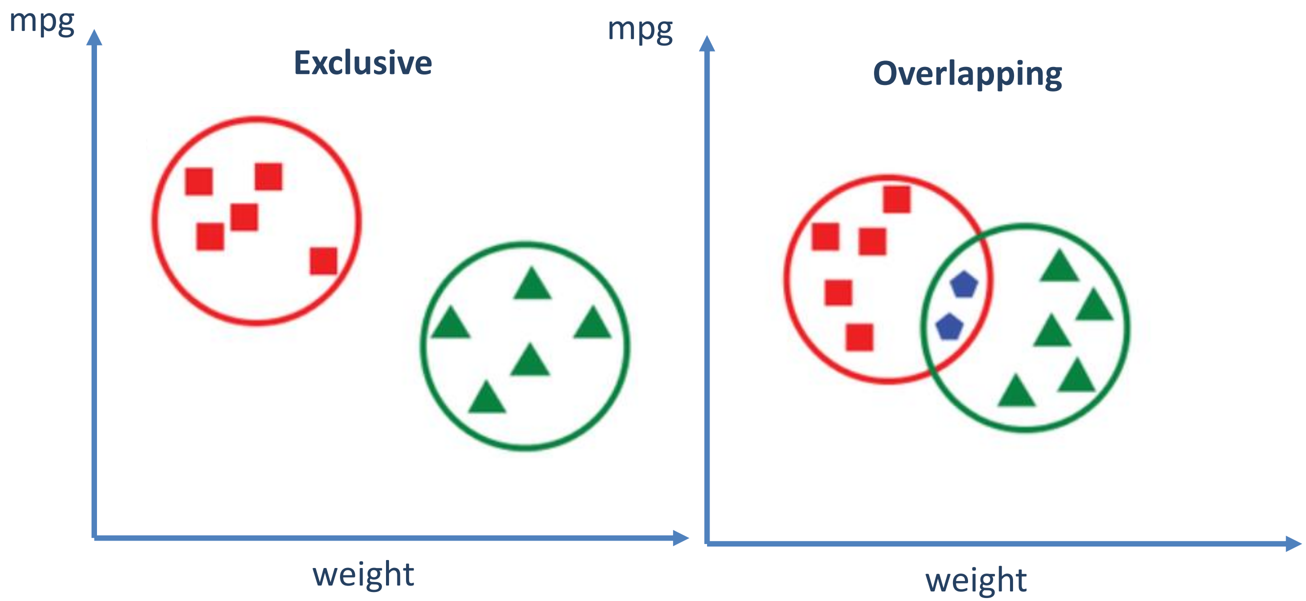
- Pre-Requisites
  - Set of samples
  - Similarity measure to compare objects
- Goal: Find groups of objects (=clusters)
- Conditions:
  - Objects within one cluster are similar to each other
  - Objects in different clusters are different from each other
- Result: Descriptive grouping of objects



Source: http://scikit-learn.sourceforge.net/0.5/_images/plot_mean_shift.png

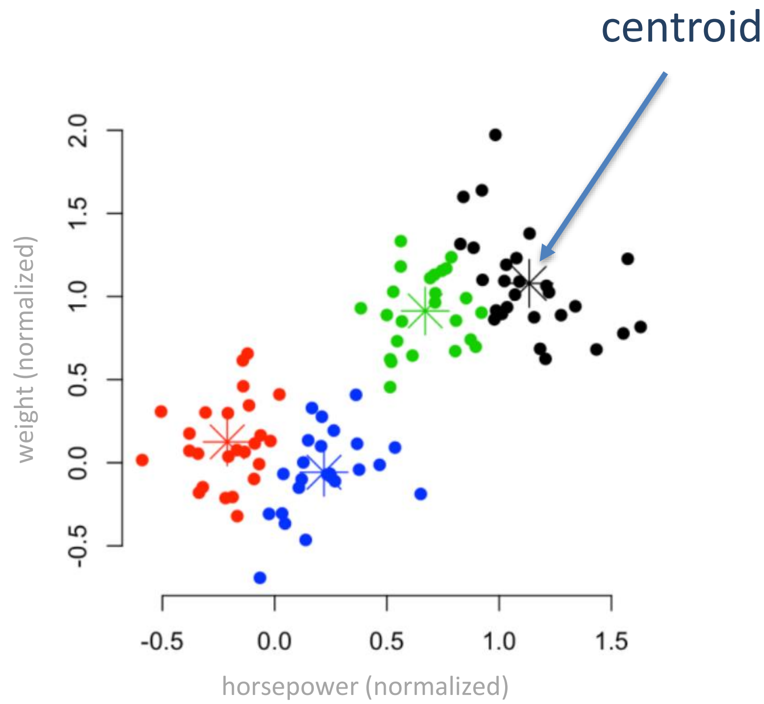
Introduction

- Basic difference of cluster analyses
  - Exclusive or strict partitioning clusters
    - Each data object belongs to one exclusive cluster (most common type of cluster analysis)
  - Overlapping clusters: Cluster groups are not exclusive and each data object may belong to more than one cluster.



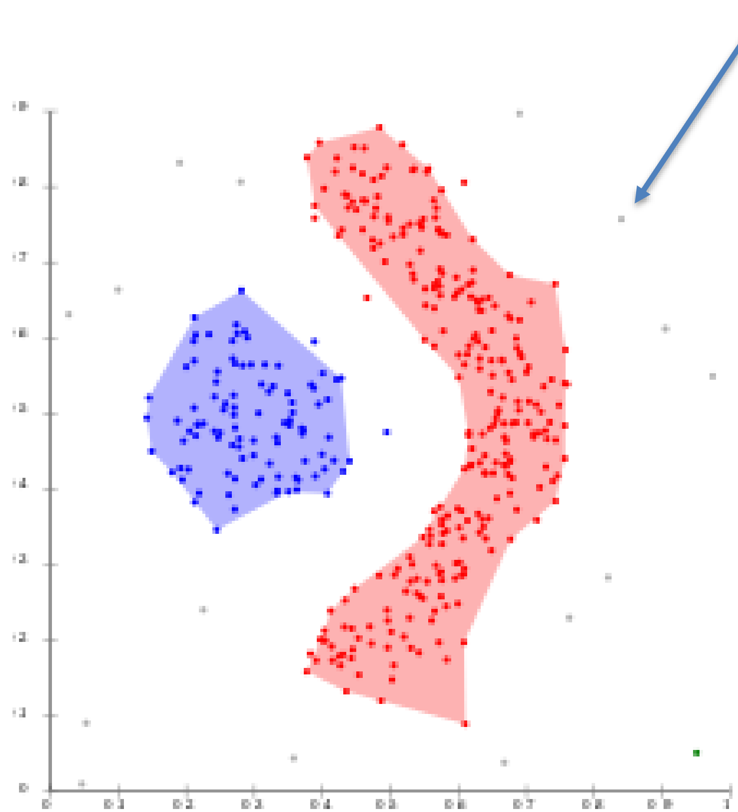
Introduction

- Prototype-based clustering
  - Each cluster is represented by a central data object, also called a prototype.
  - The prototype of each cluster is usually the center of the cluster, also called "centroid".
  - Objects with similar properties like the centroid are grouped in this cluster.



Introduction

- Density-based clustering
  - Cluster is defined as a dense region where
    - objects are concentrated surrounded by
    - a low-density area where data objects are sparse.
  - Each dense area is assigned a cluster and the low-density area can be discarded as noise.
  - Note: not all data objects need to be clustered



Source: https://en.wikipedia.org/wiki/DBSCAN

Agenda – Part 5

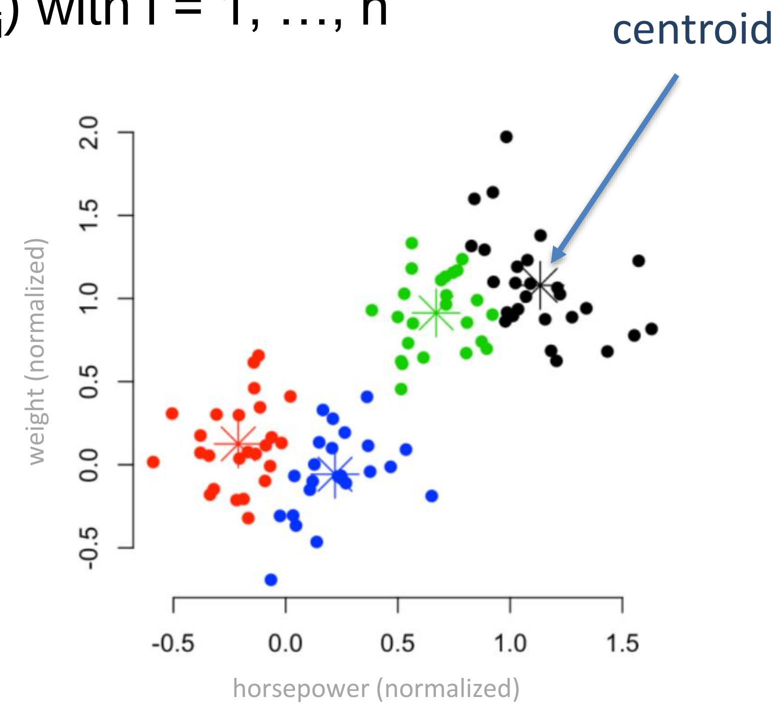
- Introduction to Unsupervised Learning and Clustering
- Prototype-based Clustering (k-Means)
- Density-based Clustering (DBSCAN)
- Summary & Outlook

k-Means Clustering

- Prototype-based method which divides data into k clusters
  - k needs to be pre-defined.
  - Goal: Find k clusters from a collection of m objects with n attributes

- Each sample can be formalized as (x_i, y_i) with i = 1, …, n
  - In case of two features x and y
  - n=number of samples in the dataset

- For a given cluster, the point that corresponds to the mean value of all samples' x and y-values called centroid
  - Variations:
    - K-Median: median value instead of mean
    - K-Medoid: most centrally located data point (out of the original samples)

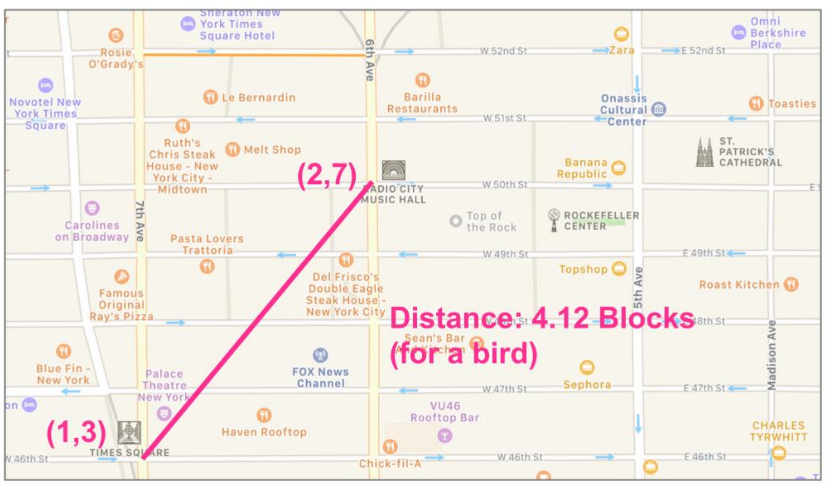


k-Means Clustering: Similarity Measures

- Remember the distance measures? These are important in k-Means clustering, too, since we need a mechanism to evaluation “proximity” of samples.

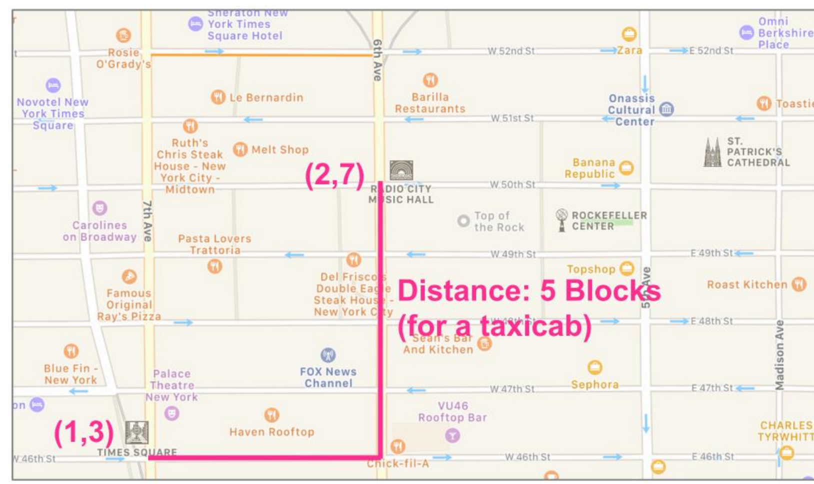
Euclidian distance

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



Manhattan distance

$$d(x, x') = \sum_{i=1}^{m} |x_i - x'_i|$$



k-Means Clustering

Algorithm

1. Choose value of k and k initial centroid guesses

2. Compute distance d_{ij} from each data point (x_i ,y_i) to each centroid (cx_j ,cy_j) and assign each point to closest centroid:

$$d_{ij} = \sqrt{(x_i - cx_j)^2 + (y_i - cy_j)^2}$$

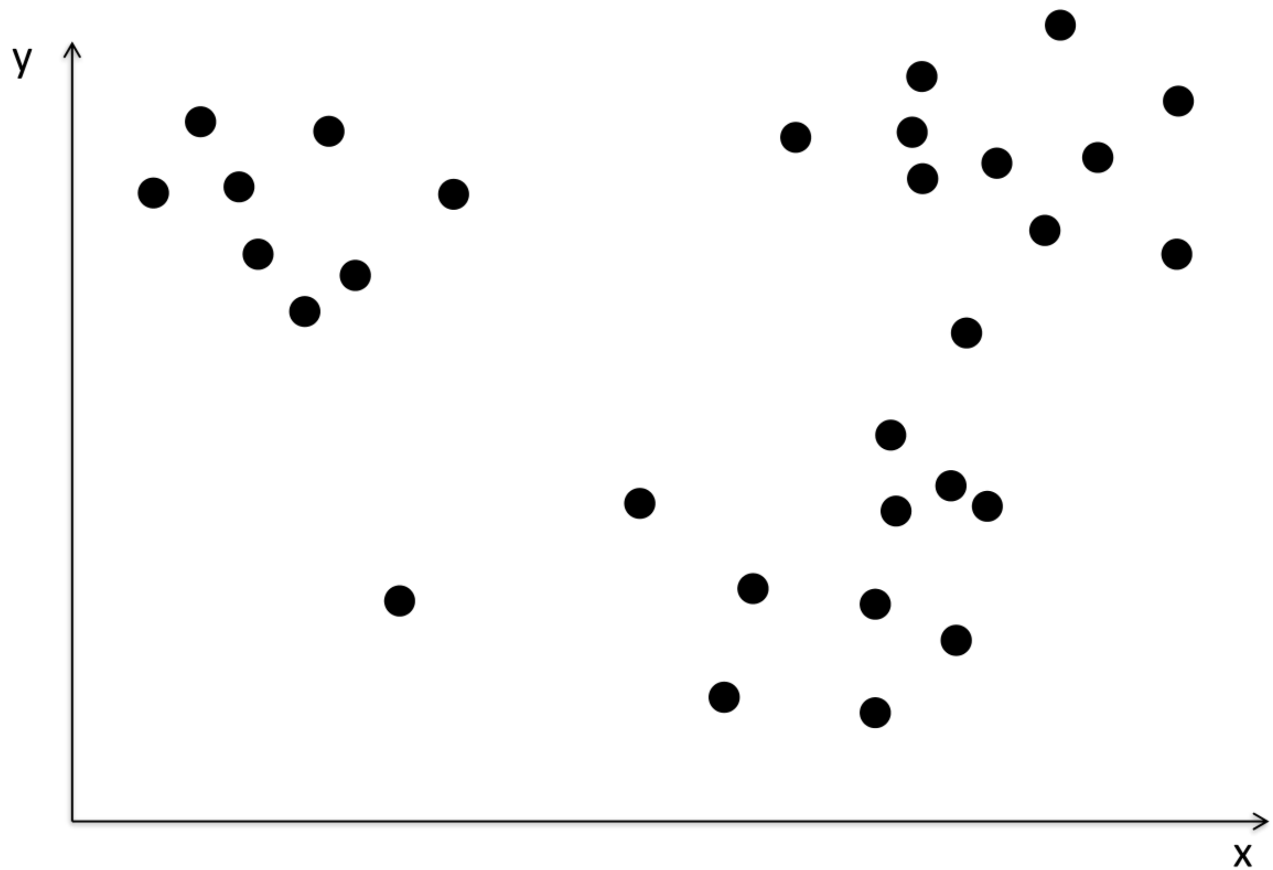
3. Compute the centroid of each defined cluster from step 2.
The centroid of the m samples in a k-means cluster is calculated as:

$$(x_i, y_i) = \left(\frac{\sum_{i=1}^{m} x_i}{m}, \frac{\sum_{i=1}^{m} y_i}{m}\right)$$

4. Repeat steps 2 and 3 until centroids stay stable

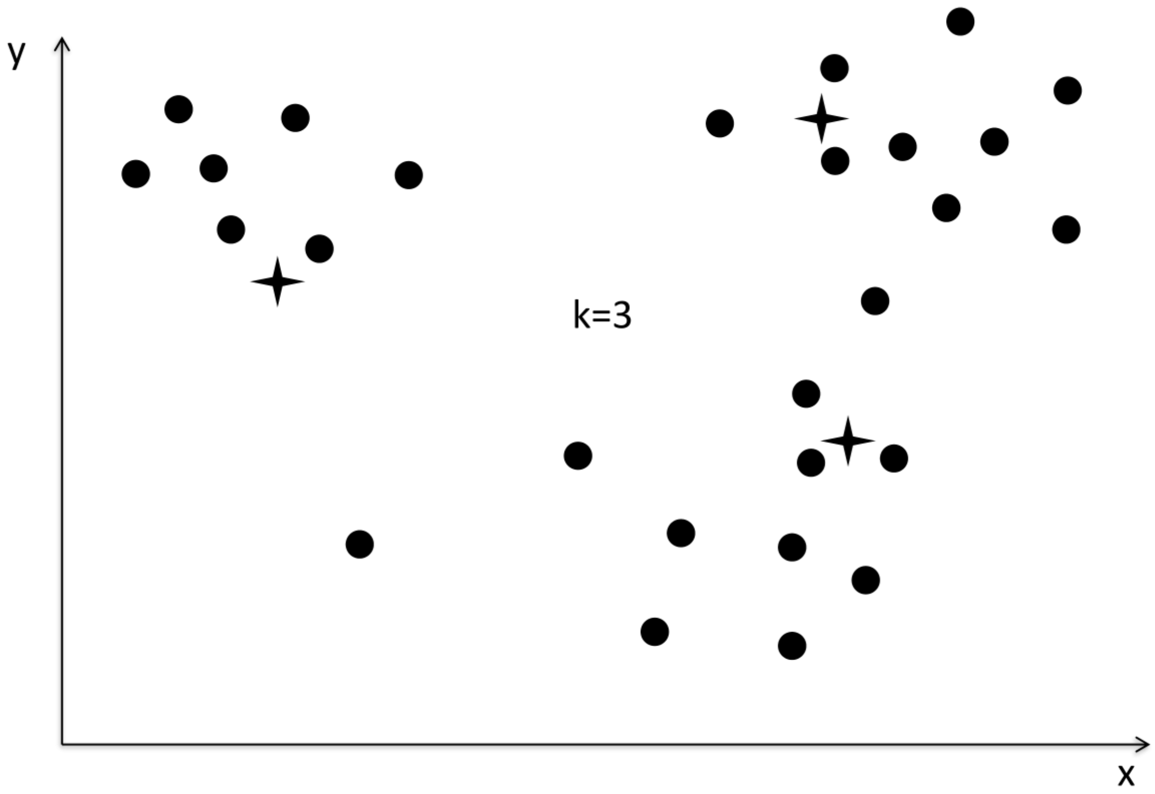
k-Means Clustering: Example

- Initial samples with two features: x and y:



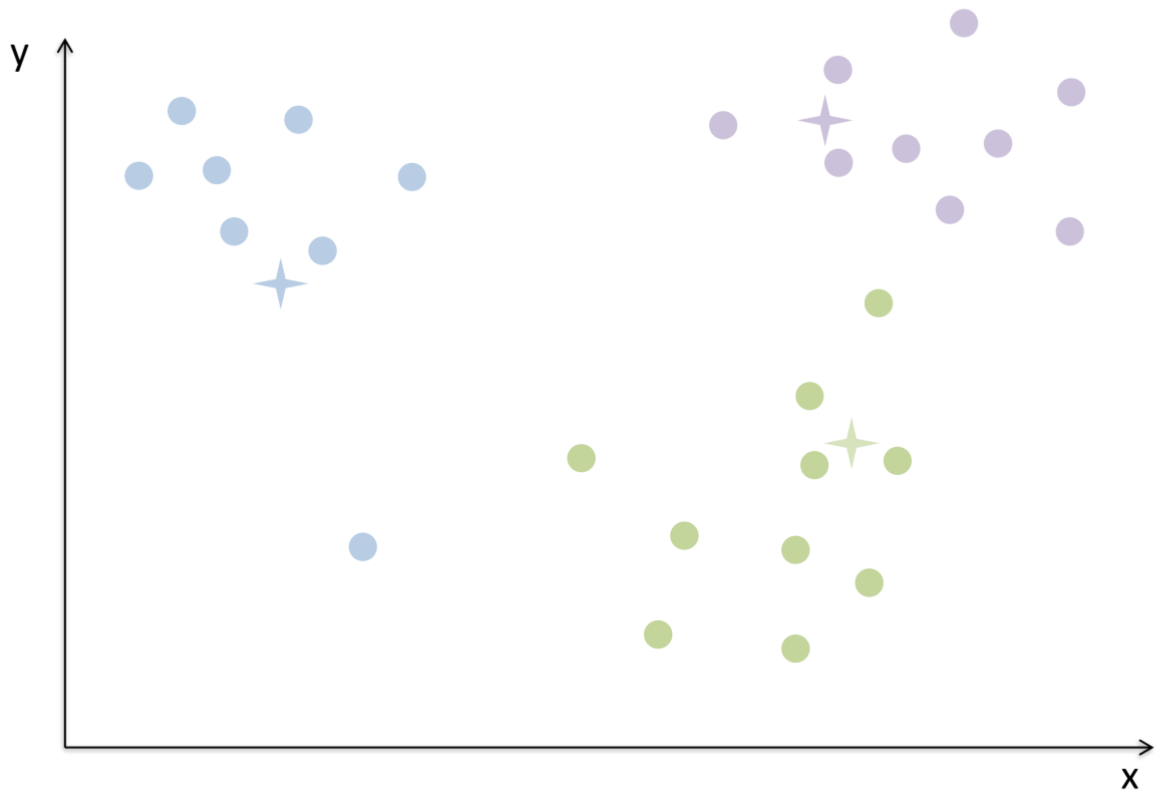
k-Means Clustering: Example

- Step 1: Choose value of k and guess initial centroids



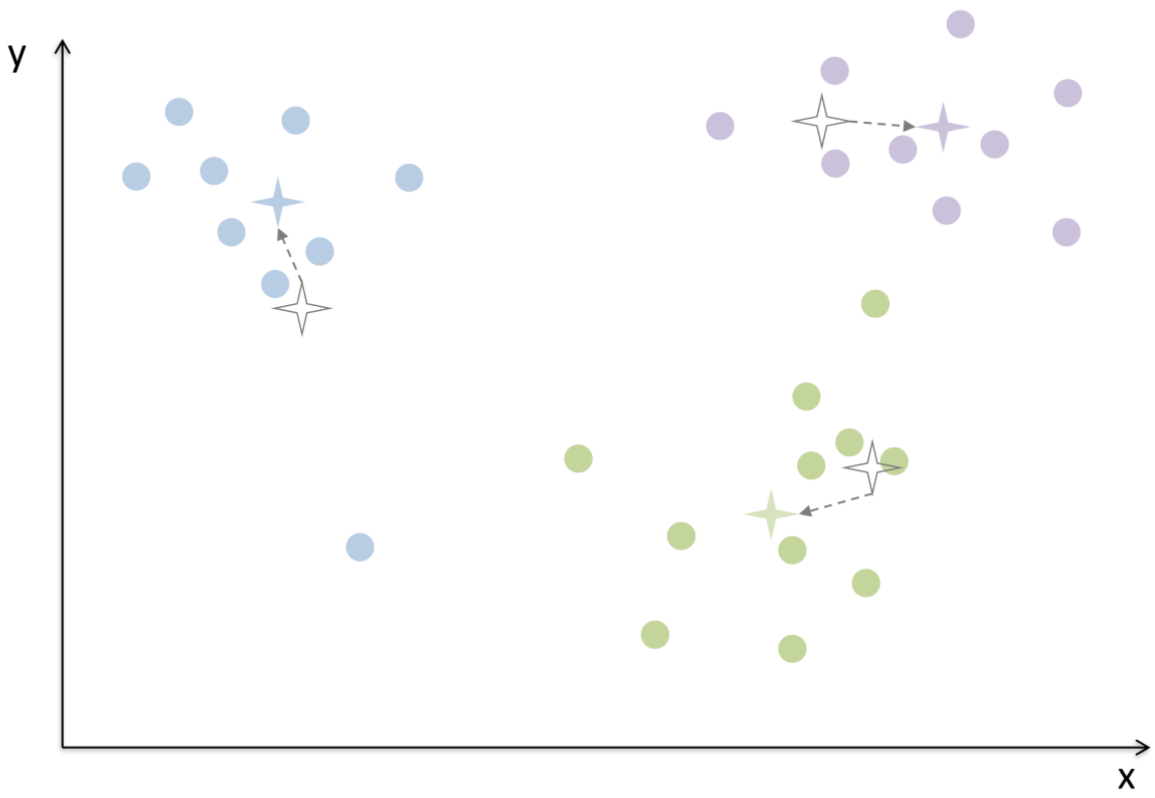
k-Means Clustering: Example

- Step 2: Compute distance of each data point to each centroid and assign each point to its closest centroid



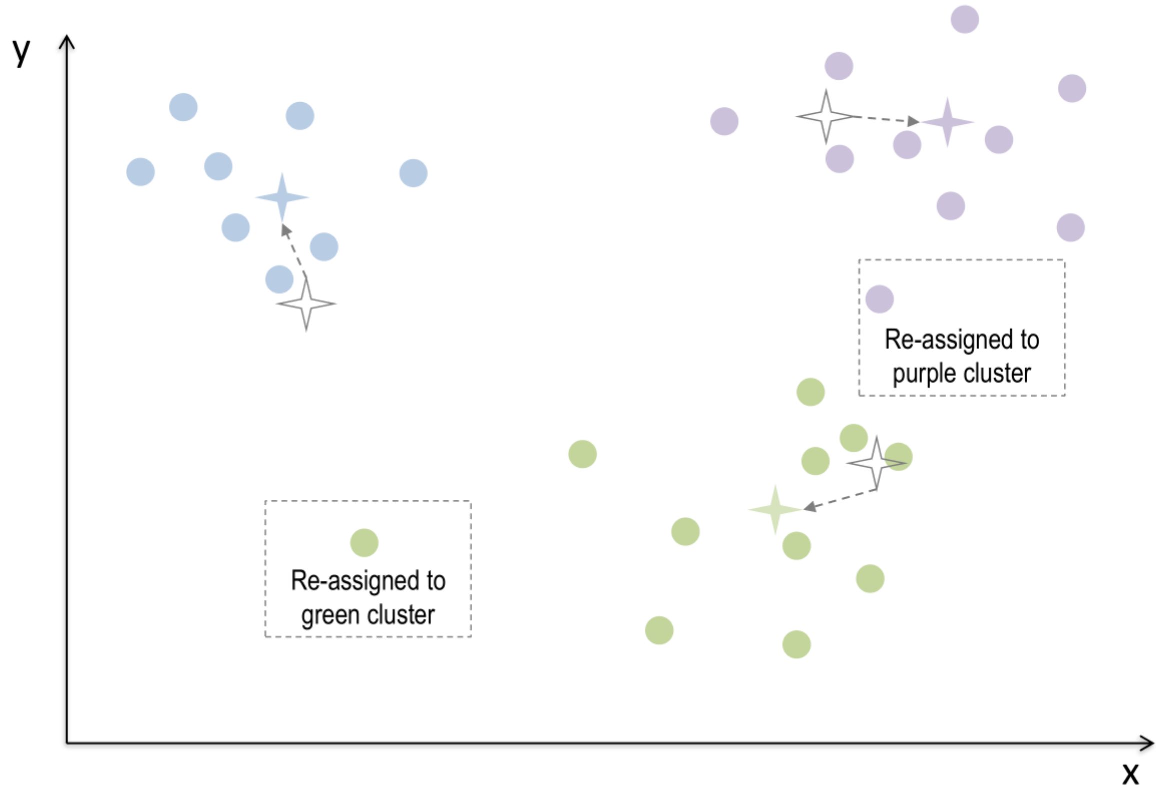
k-Means Clustering: Example

- Step 3: Compute the centroid of the newly defined clusters by calculating the mean for x and y



k-Means Clustering: Example

- Step 4: Repeat steps 2 and 3 until centroids stay stable:
  Re-assign each point to closest centroid & compute the centroid of each newly defined cluster

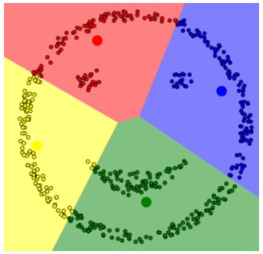


k-Means Clustering: Example

- Let us take a look at another example:

  https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

- *What are the challenges?*
  - Number of centroids (k)
  - Initial centroid position influences clustering result
  - k-Means cannot distinguish dense areas and hence identify arbitrary types of clusters.



Lab

- Clustering using sklearn

k-Means Clustering: Finding the right k

Clustering with small SSE

Clustering with large SSE

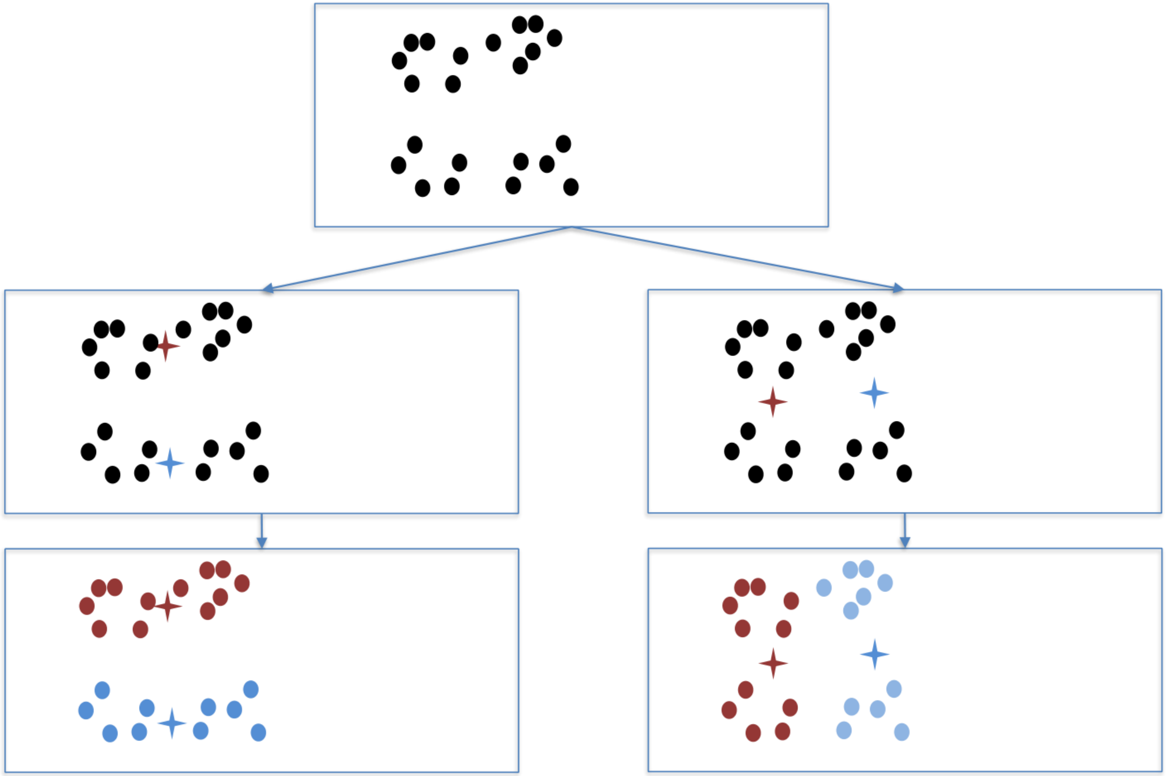
- Challenge: finding the best value for k
- We can use Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest centroid.
  - To get the SSE, we square these errors and build the sum.

$$\text{SSE} = \sum_{i=1}^{M} d(p_i, cpi)^2 = \sum_{i=1}^{M} (p_i - cpi)^2$$

  - p_i: one sample of M data points in total
  - cp_i: centroid to which p_i is associated
  - We should prefer the k which results in the smallest SSE.
- Another measure for cluster-quality is the Davies-Boulding index.
  - Measure of uniqueness of the clusters
    - Considers cohesiveness of the cluster (distance between the data points and center of the cluster) and separation between the clusters
  - The lower the value of the Davies-Bouldin index, the better the clustering.

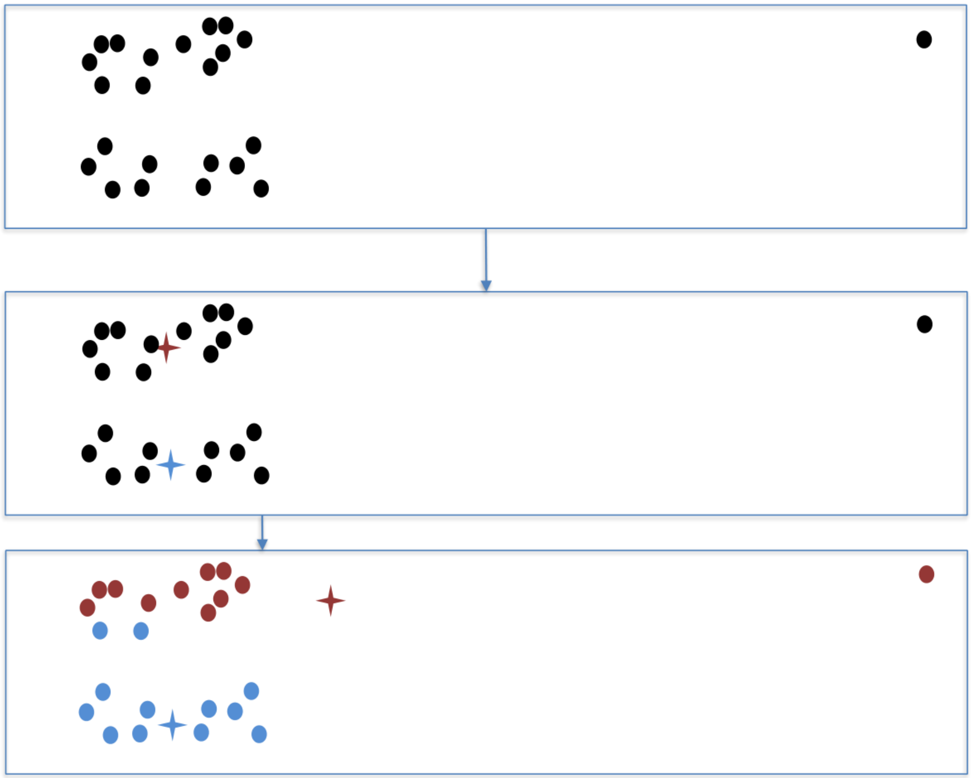
k-Means Clustering: Weaknesses

- Importance of Initial Centroids:



k-Means Clustering: Weaknesses

- Influence of Outliers:

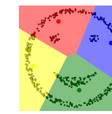


Agenda – Part 5

- Introduction to Unsupervised Learning and Clustering
- Prototype-based Clustering (k-Means)
- **Density-based Clustering (DBSCAN)**
- Summary & Outlook

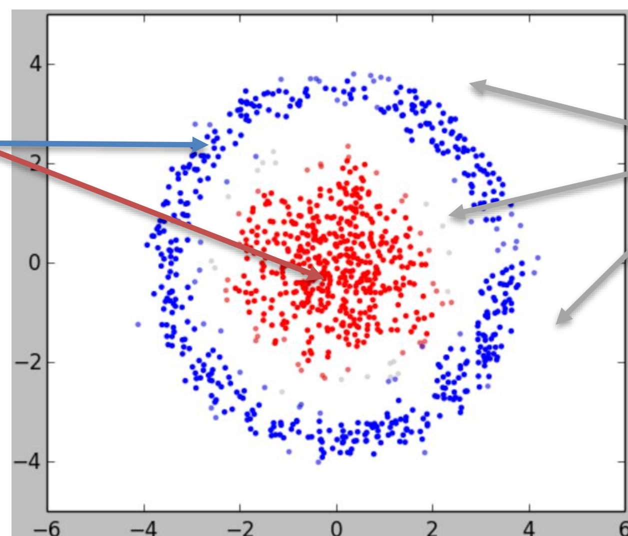
Density-Based Clustering: Introduction

- In many applications, the number of clusters will not be known a priori.
  - → Issue of k-Means
- Furthermore, we saw that k-Means cannot handle complex data structures that we might need to handle.



- For these situations, we can rely on density-based clustering:
  - Clusters are formed by high-density areas amongst relatively low-density areas.
- Example

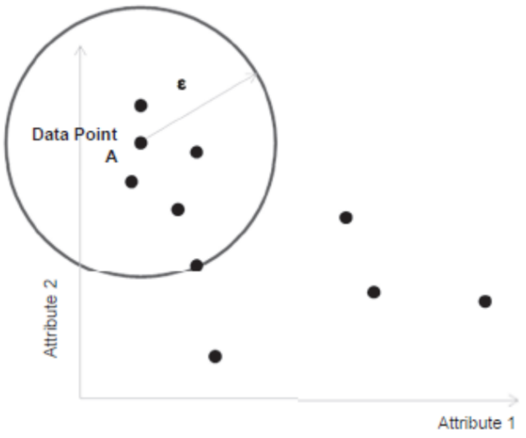
High-density areas



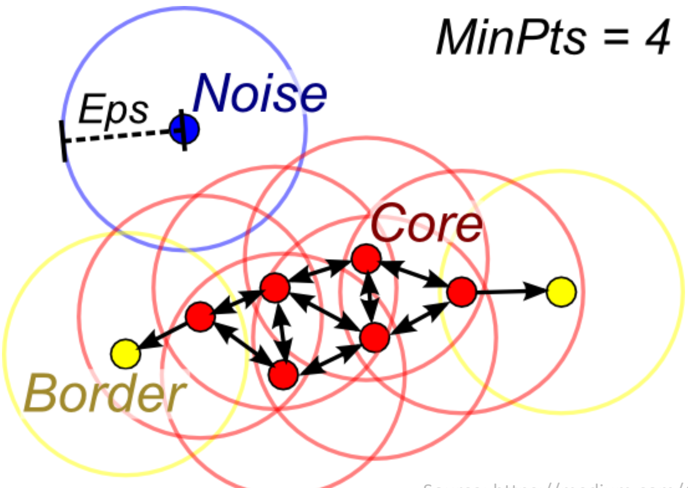
Low-density areas

Density-Based Clustering: Introduction

- Measuring density
  - Density = number of points within a circular space with radius ε (epsilon)
  - Example: Density around data point A is 6



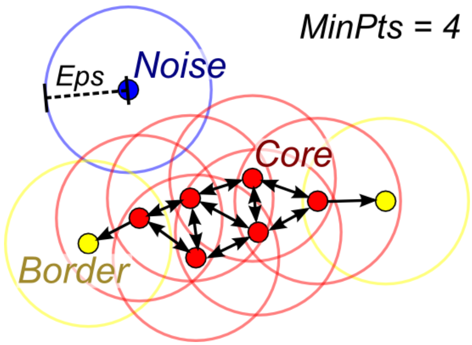
- Based on this idea, the DBSCAN algorithm identifies dense regions based on the hyperparameters radius (ε) and the minimum number of points that are necessary to define a region dense (MinPts):



Source: https://medium.com/@elutins/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818

Density-Based Clustering: DBSCAN

- DBSCAN "walks" through the whole dataset and calculates the density for all data points with a given radius ε (epsilon).
  - Any density above MinPoints is considered high density.



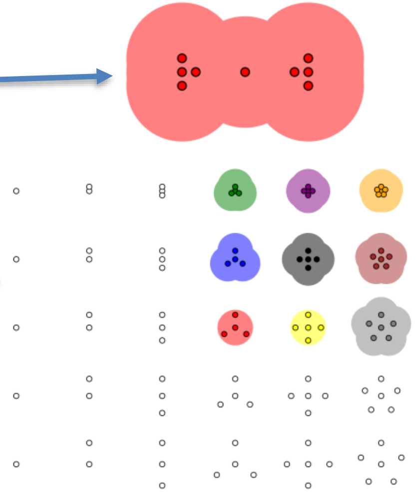
- Using a pre-defined threshold (MinPoints), DBSCAN decides whether a **neighborhood is high density or low density**
  - DBSCAN distinguishes core points and border points (forming high density areas).
    - A interconnected high density area becomes one cluster!
  - Points that are not in a dense region (or at the border of a dense region)
    - Labeled as "Noise" (default cluster)

Source: https://medium.com/@elutins/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818

Density-Based Clustering: DBSCAN

- Let us take a look at how this algorithm works: https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

- Issues
  - If the dataset contains regions with varying density, DBSCAN is not able to identify the clusters correctly.
    - Either we set MinPoints too low / epsilon too high
      → worst case: identify just one cluster
    - Or we set MinPoints too high / epsilon too low
      → worst case: assign all points as "Noise", i.e. do not "jump" over low-density regions



  - K-means clustering is more suitable in this case

Source: https://medium.com/@elutins/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818

Density-Based Clustering: DBSCAN

- Advantages
  - Does not require specification of number of clusters in the data a priori, as opposed to k-means.
  - Can find arbitrarily shaped clusters
  - Simplicity: Requires just two parameters
- Disadvantages
  - Risk of finding bridges between two natural clusters and merging them into one cluster
  - Does not work well if data has varying densities
  - Not suitable if all data points shall be assigned to a cluster
  - Data sets with high number of attributes will have processing challenges with density clustering

Overall Conclusion / Recommendation

- In many practical applications, the number of clusters to be discovered will be unknown
  - DBSCAN can work better here than k-Means clustering
  - MinPts and ε should be set by a domain expert (data needs to be well understood)

- Given the complementary pros and cons of the k-Means and DBSCAN methods, it is advisable to cluster the data set by **both methods** and understand the patterns of both result sets

→ Next step would be Ensemble Learning (apply both methods and implement a voting mechanism)
We will not go into detail on this!

Agenda – Part 5

- Introduction to Unsupervised Learning and Clustering
- Prototype-based Clustering (k-Means)
- Density-based Clustering (DBSCAN)
- Summary & Outlook

Thank you!