Lecture 11 – Clustering

Amit K Verma, Anthony (Tony) Rollett

Carnegie Mellon University

Overview

- Clustering
    - Dissimilarity Index
    - Examples of High Dimensional Data
    - Hard Clustering Algorithms
        - *Hierarchical*
        - *K-means*
        - *DBSCAN*
        - *t-SNE*

Source: Machine Learning for Materials Research Bootcamp & Workshop on Machine Learning Microscopy Data - https://www.nanocenter.umd.edu/events/mlmr-2020/

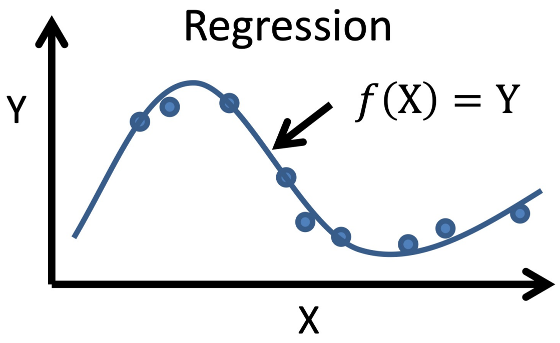
- *Clustering* by Dr. A Gilad Kusne, NIST – MD (2018)

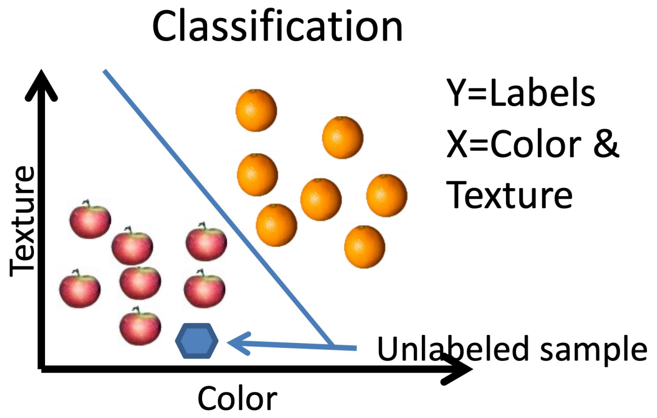
Supervised Learning

- Find the function f that maps the input data x to the output data y

$$f: x \rightarrow y$$

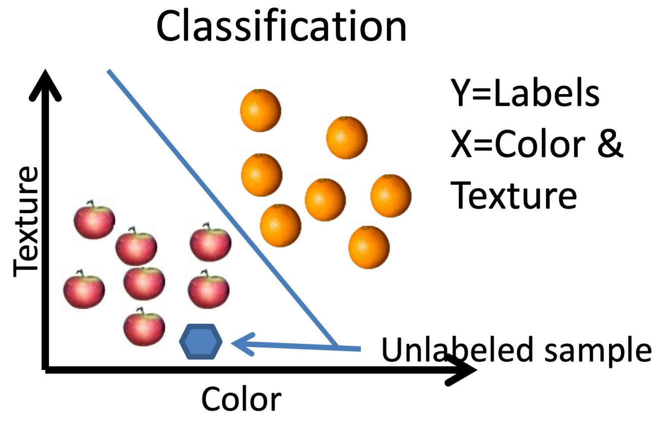
- y is continuous: Regression
- y is discrete: Classification
- Cross-validation to check performance & determine parameters

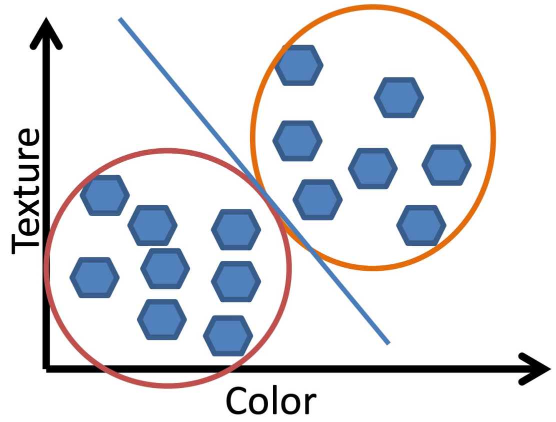




Unsupervised Learning

- “Unsupervised”: We don’t have output data y (“learning without a teacher”)
- Interested in relationship between the data x
- Learn about x from its distribution
- Cross-validation for algorithm performance isn’t available
    - Performance checked with: Heuristics & Expert analysis

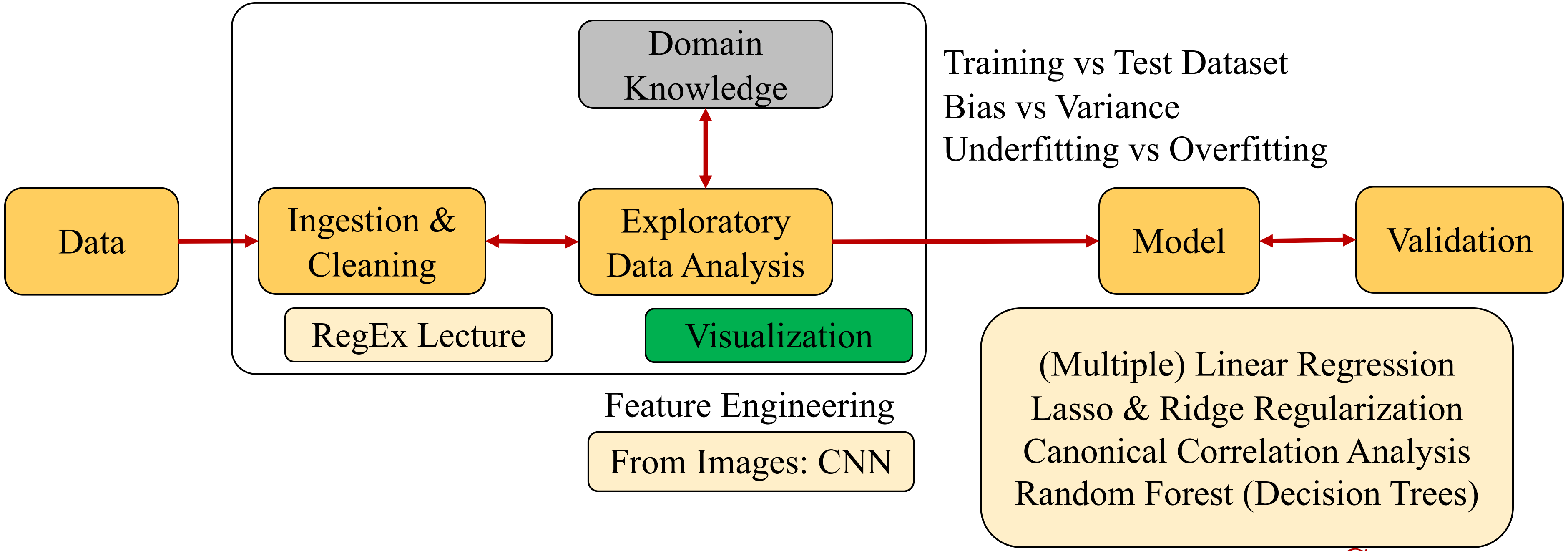




Note: Dimension reduction is the most common application of unsupervised learning

Clustering

Where does it fit


Domain Knowledge
Data
Ingestion & Cleaning
Exploratory Data Analysis
RegEx Lecture
Visualization
Training vs Test Dataset
Bias vs Variance
Underfitting vs Overfitting
Model
Validation
Feature Engineering
From Images: CNN
(Multiple) Linear Regression
Lasso & Ridge Regularization
Canonical Correlation Analysis
Random Forest (Decision Trees)

Carnegie Mellon University

Clustering

- An unsupervised learning technique
- Assignment of a set of observations into subsets (called clusters), such that those within each cluster are more closely related than objects assigned to different clusters
- Dissimilarity Measures: quantify the dissimilarity or difference between two samples,

$$d\ (x1,\ x2) \leftrightarrow K\ (x1,\ x2)$$

- “Specifying an appropriate dissimilarity measure is far more important in obtaining success with clustering than choice of clustering algorithm.” – Hastie, et al.
- **Need domain knowledge!**

Carnegie Mellon University

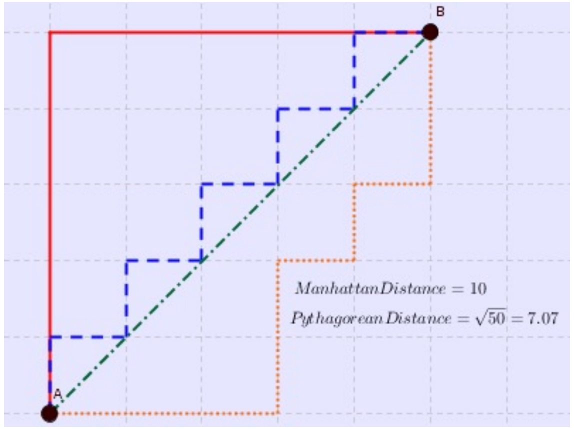
Dissimilarity Measures

- Euclidean (most common) (L2): $d_E(\boldsymbol{x}_1, \boldsymbol{x}_2) = \left[\sum_i (x_{1i} - x_{2i})^2\right]^{1/2}$

- Taxi-Cab (L1): $d_T(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_i |x_{1i} - x_{2i}|$

- p-norm : $d_p(\boldsymbol{x}_1, \boldsymbol{x}_2) = \left[\sum_i |x_{1i} - x_{2i}|^p\right]^{1/p}$

- 1-3 dimensions


B
A
$Manhattan Distance = 10$
$Pythagorean Distance = \sqrt{50} = 7.07$

- scipy documentation: https://docs.scipy.org/doc/scipy/reference/spatial.distance.html
  - Many types: geometric, statistical (in RF), information theory, feature preserving, etc.

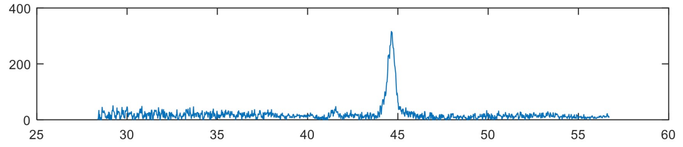
- Algorithms are **highly sensitive to scales**
  - Common practice to normalize / standardize the features

Carnegie Mellon University

High Dimensional Data

- Intensity as a function of frequency, angle, location, etc.

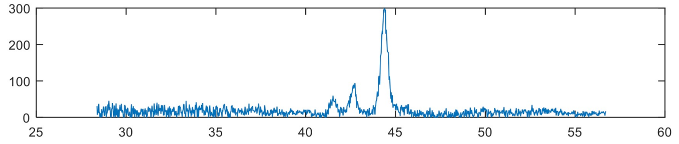
Scale Invariant (ignore height)



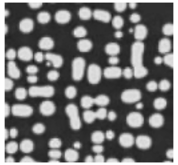
$$\boldsymbol{x}_1 = [0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 4\ 5\ 7\ 10\ 11\ 9\ 8\ 4\ 3\ 1\ 0\ 0\ 0 \ldots]$$

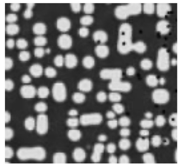
$$\xleftrightarrow{\quad D \quad}$$

Translation invariant (ignores shifts)



$$\boldsymbol{x}_2 = [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 4\ 5\ 7\ 10\ 11\ 9\ 8\ 1\ 0\ 0\ 0\ \ldots]$$



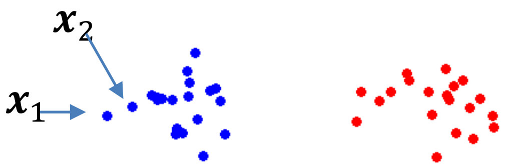


$$\boldsymbol{x}_1 = \begin{bmatrix} 0&1&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0 \\ 1&1&1&0&0&0&1&0 \\ 1&4&1&0&0&0&0&0 \\ 1&2&1&0&0&0&0&0 \end{bmatrix}$$

$$\boldsymbol{x}_1 = [0\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 4\ 2\ 0\ 0\ 1\ 1\ 1\ \ldots]$$

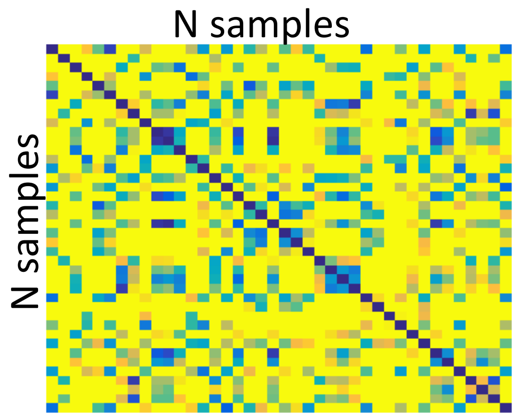
Dissimilarity Matrix

Data from two Gaussian distributions, labelled by color

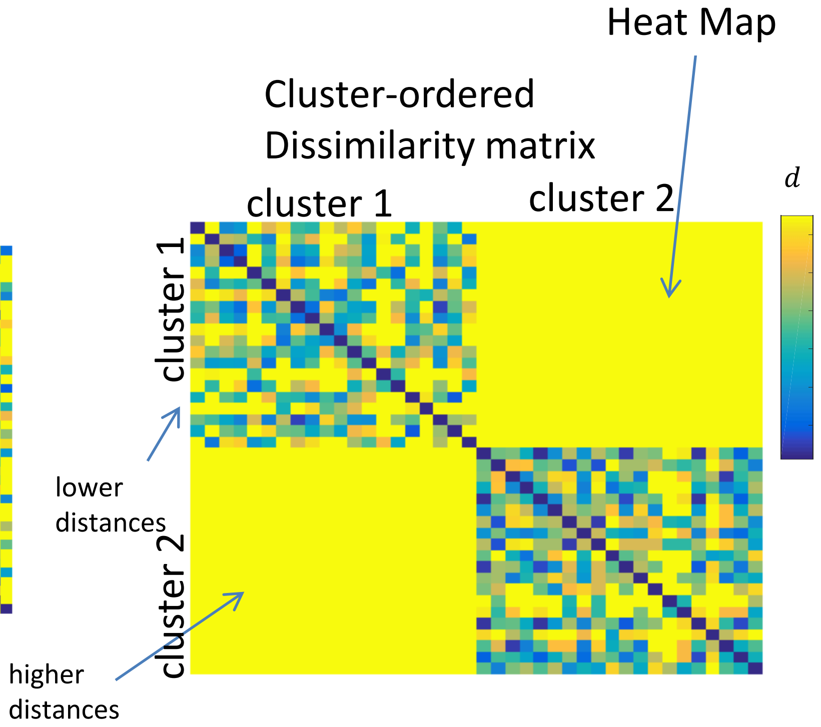


$$\begin{array}{c|cccc} & x_1 & x_2 & x_3 & x_4 \\ \hline x_1 & & & & \\ x_2 & & & & \\ x_3 & & & & \\ x_4 & & & & \end{array}$$

Unordered Dissimilarity matrix $d(x_i, x_j)$

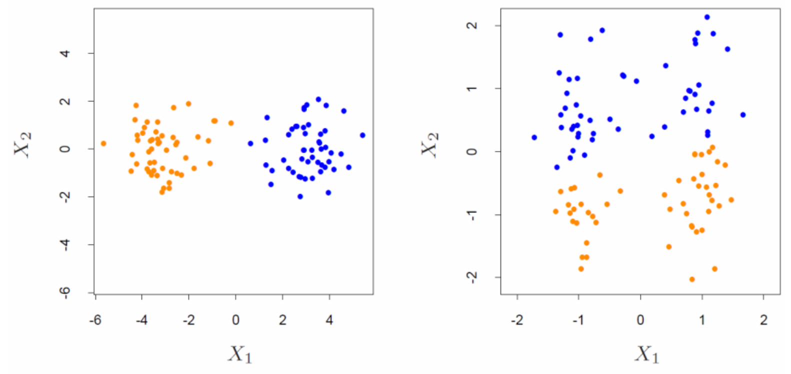


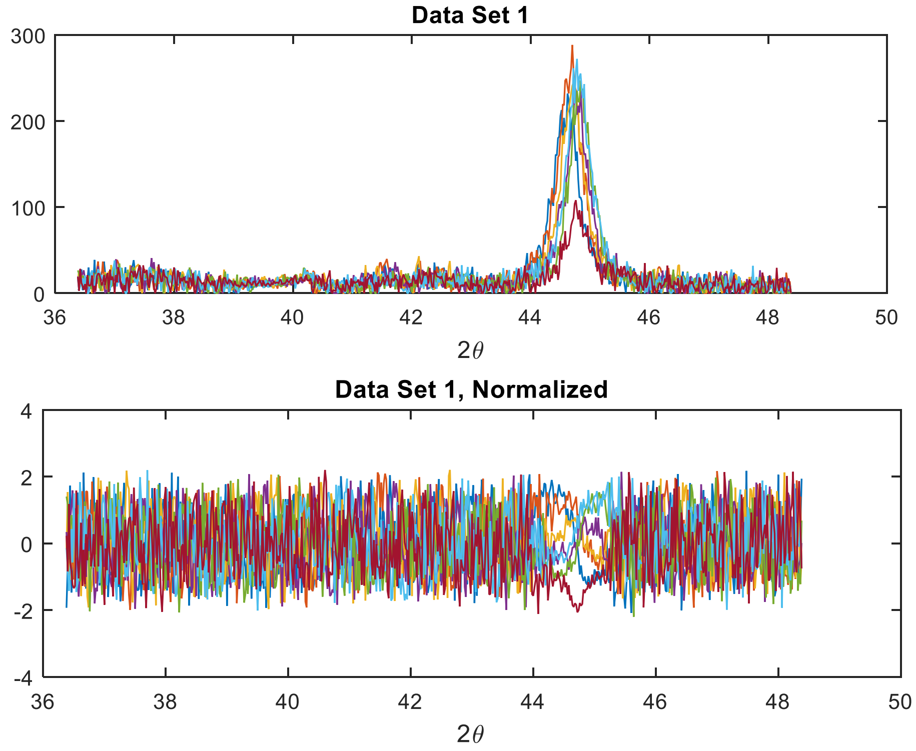
Cluster-ordered Dissimilarity matrix



Dissimilarity (/similarity) matrix is at the core of unsupervised learning

Data Normalization: Issues





Normalizing data may warp the distribution and make analysis difficult

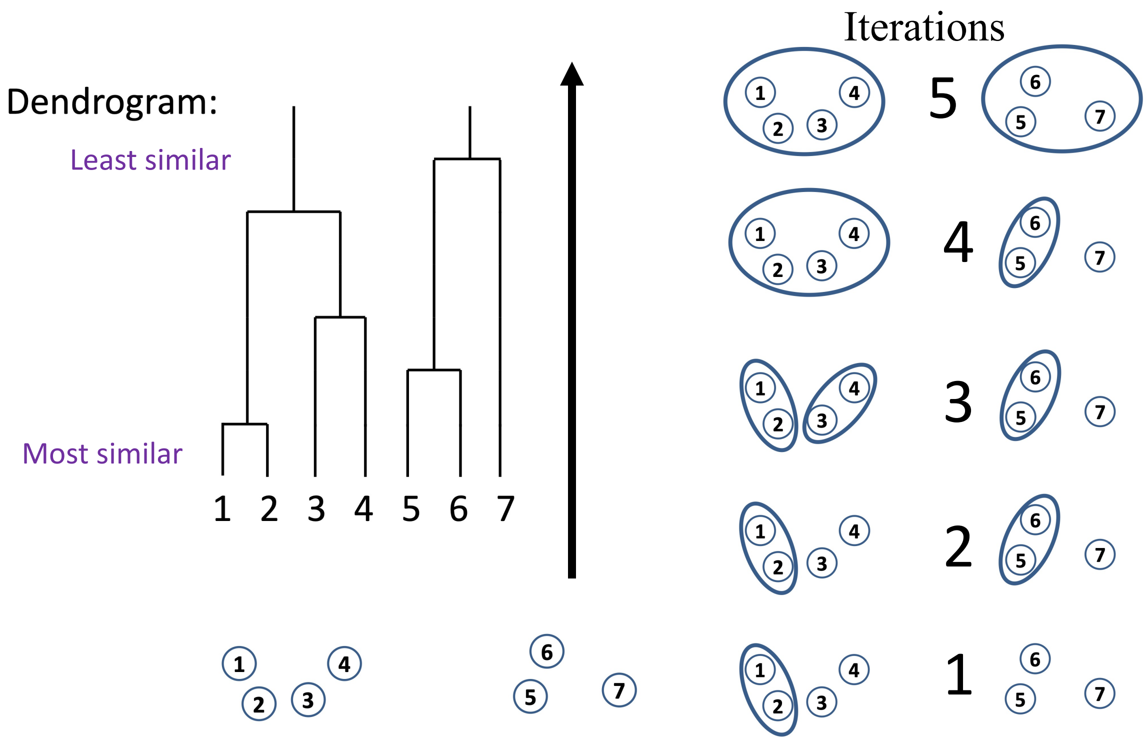
Classification of (Hard) Clustering Algorithms

- Hierarchical Algorithms: find successive clusters using previously established clusters
    - agglomerative ("bottom-up") : begin with each element as a separate cluster and merge them into successively larger clusters
    - divisive ("top-down") : begin with the whole set and proceed to divide it into successively smaller clusters

Agglomerative HC

- Don't need to define the number of clusters
- Need dissimilarity matrix to define dissimilarity between clusters

1. We start with assigning each observation to its own cluster
2. Then, compute the similarity (or distance) between each of the clusters
3. Join the two most similar clusters
4. Finally, repeat steps 2 and 3 until there is only a single cluster left



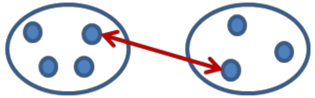
Carnegie Mellon University

(Few) Measures of Similarity (/Distance)

s and **t** are combined to form cluster **u**. Let **v** be any remaining cluster in the forest that is not **u**. The following are methods for calculating the distance between the newly formed cluster **u** and each **v**.

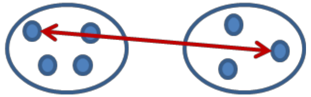
- method = 'single' assigns

  d(u,v) = min (*dist(u[i], v[j])*); for all points *i* in cluster **u** and *j* in cluster **v**



- method='complete' assigns

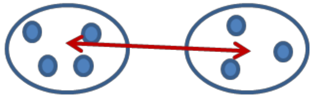
  d(u,v) = max(*dist(u[i], v[j])*); for all points *i* in cluster **u** and *j* in cluster **v**



- method='average' assigns

$$d(u,v) = \sum \frac{d(u[i],v[j])}{(|u| * |v|)};$$

for all points *i* and *j* where |u| and |v| are the cardinalities of clusters u and v, respectively



Carnegie Mellon University

Classification of (Hard) Clustering Algorithms

- Hierarchical Algorithms: find successive clusters using previously established clusters
    - agglomerative ("bottom-up") : begin with each element as a separate cluster and merge them into successively larger clusters
    - divisive ("top-down") : begin with the whole set and proceed to divide it into successively smaller clusters

- Partitional Algorithms: determine all clusters at once (k-means); require the number of clusters (X-means: automatically determine the number of clusters based on BIC scores; G-means)

K-means Algorithm

- Number of clusters k is known
- Initialize the prototype for each cluster – often picked randomly

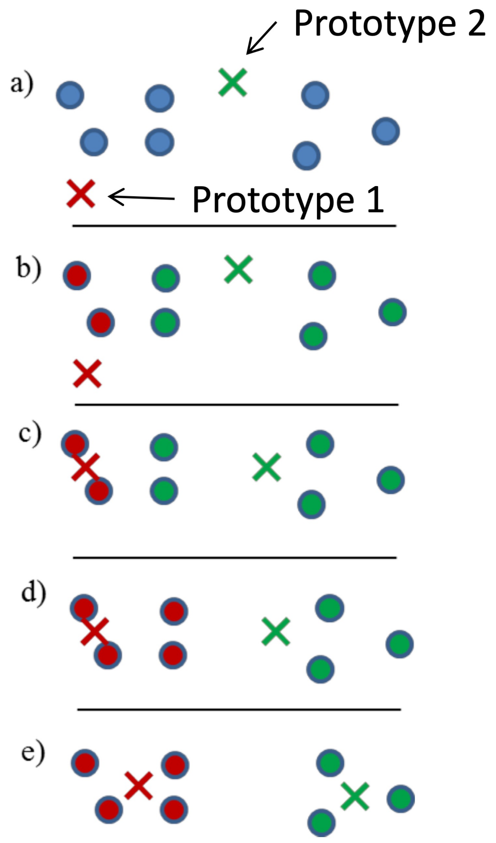
1. Assign each point to the nearest prototype – defining clusters
2. Compute the mean for each cluster.
    - prototype = mean

- Iterate (1) and (2) until convergence
- Minimizes within cluster scatter

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k} \|x_i - \bar{x}_k\|^2$$

Prototype k

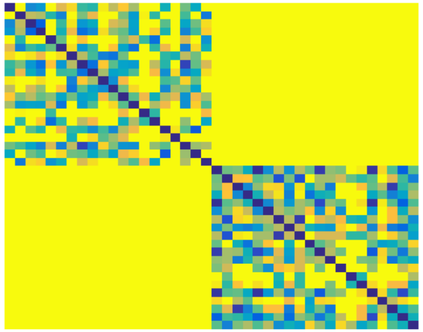
- Finds local minima. Repeat & choose result with lowest W



University

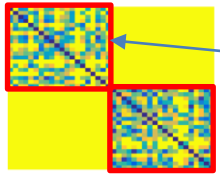
Number of Clusters?

- Classification
  - Can use training data to evaluate performance
  - Pick classification parameters that optimize performance
- Clustering
  - No training data
  - Performance is often subjective – requires expert's eye
- How to evaluate clustering results?
  - Combinatorial clustering methods (based on data itself)
    - *T = sum of all distances between methods* = $\sum_{i,j} d(x_i, x_j)$
    - *T = W(C) + B (C)*



Dissimilarity Matrix for Clustered points!

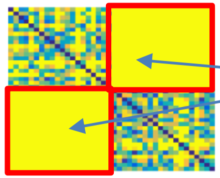
$W(C)$



Within-Cluster dissimilarities

Minimize

$B(C)$



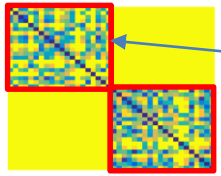
Between-Cluster dissimilarities

Maximize

Number of Clusters?

- How to evaluate clustering results?
  - Combinatorial clustering methods (based on data itself)
    - *T = sum of all distances between methods =* $\sum_{i,j} d(x_i, x_j)$
    - *T = W(C) + B (C)*

$W(C)$



Within-Cluster dissimilarities

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\|x_i - \underset{\uparrow}{\bar{x}_k}\|^2$$

Prototype k

Minimize

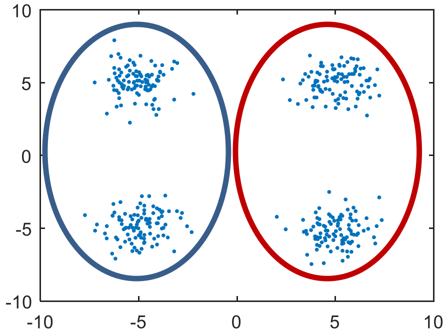
- Repeat clustering N times and return the result with minimum W(C)

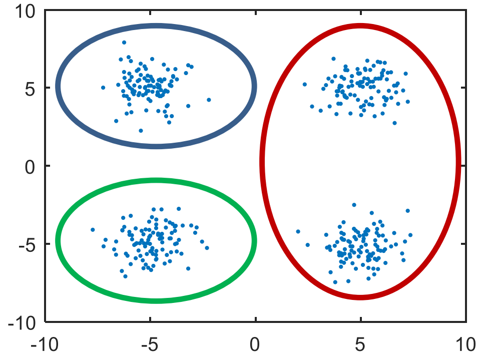
Carnegie Mellon University

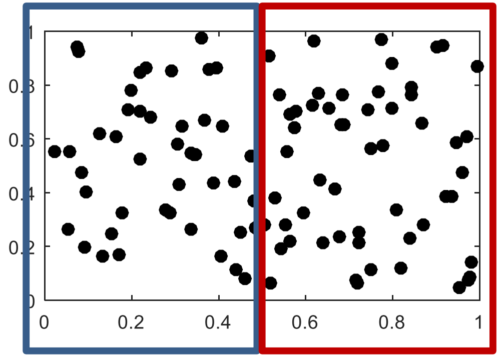
Number of Clusters?

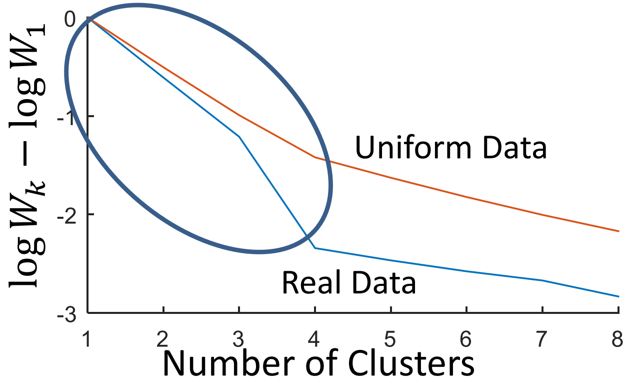
- Gap Statistic
  - Heuristic based on within cluster scatter W_k

- $\hat{k} < k$
- As $\hat{k}$ increases, more cluster centers -> W_k decreases
- Clustering becomes more accurate -> W_k decreases rapidly







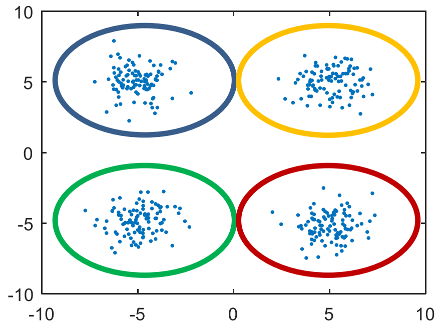


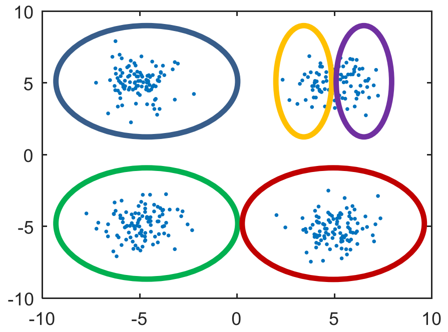
University

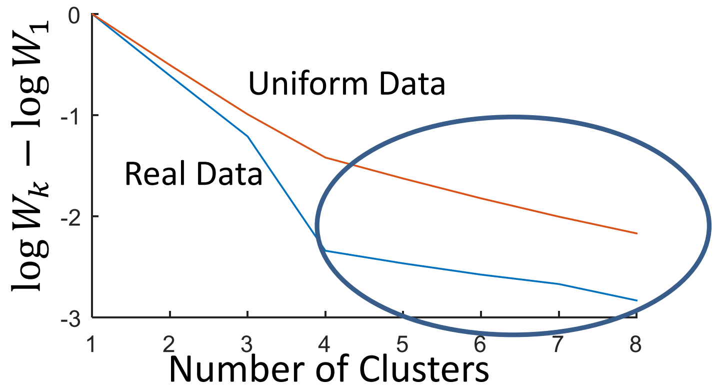
Number of Clusters?

- Gap Statistic
  - Heuristic based on within cluster scatter W_k

- $\hat{k} < k$
- As $\hat{k}$ increases, more cluster centers -> W_k decreases
- Clustering becomes more accurate -> W_k decreases rapidly
- Clustering is less accurate -> W_k decreases slower

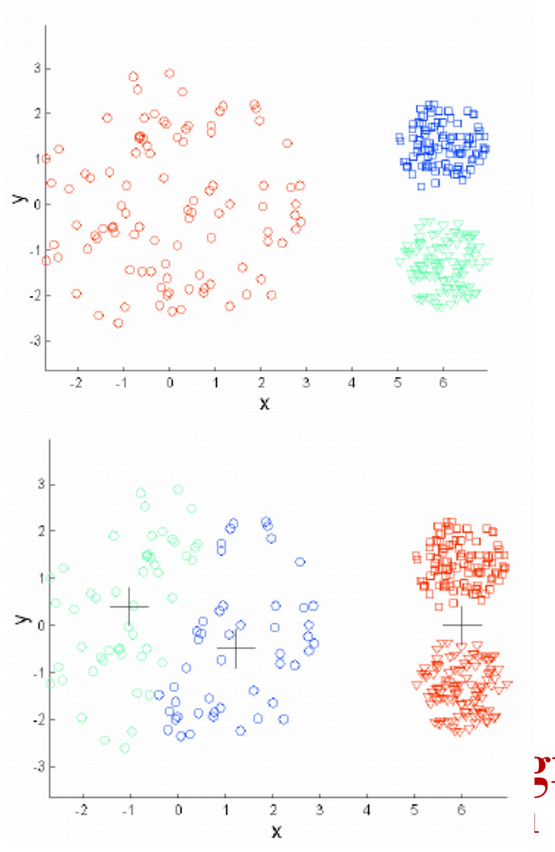






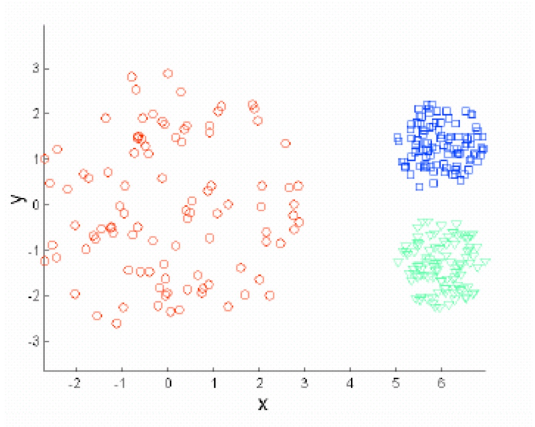
Issues with K-means Clustering

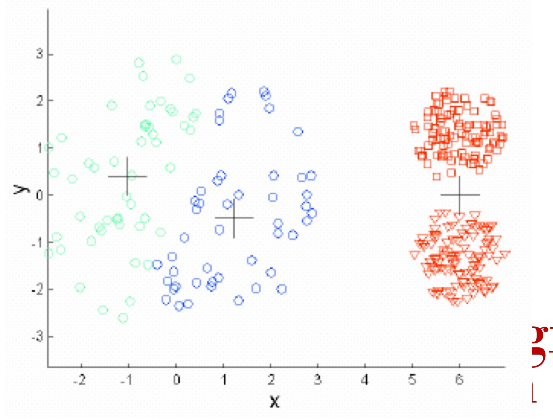
- Different cluster density
- Different cluster size

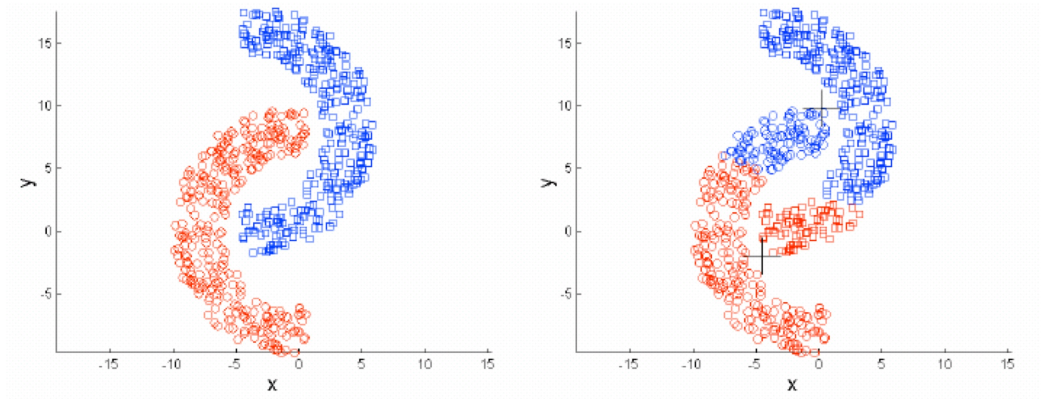


Issues with K-means Clustering

- Different cluster density
- Different cluster size
- Non-spherical Cluster



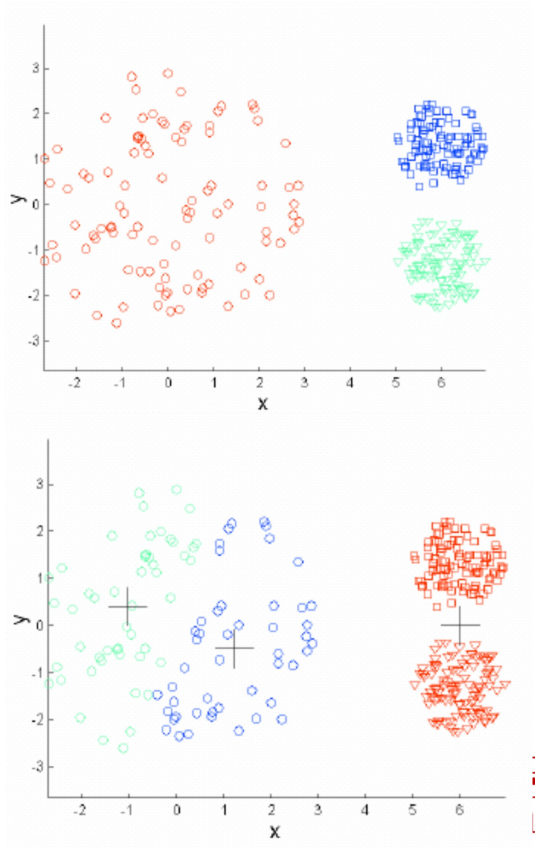


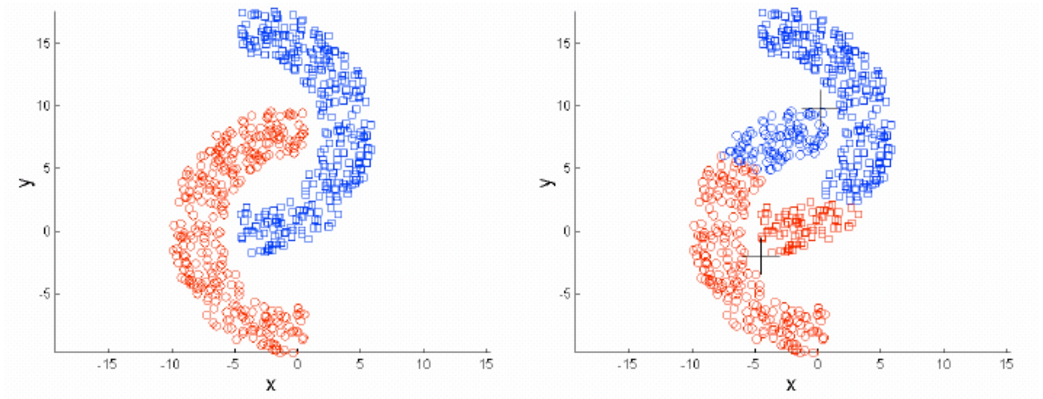


Issues with K-means Clustering

- Different cluster density
- Different cluster size
- Non-spherical Cluster
- Outliers
- Empty Cluster (e.g., due to sampling)

- Some, can be addressed using <u>Spectral Clustering</u>
    - Graph-based clustering





Classification of (Hard) Clustering Algorithms

- Hierarchical Algorithms: find successive clusters using previously established clusters
    - agglomerative ("bottom-up") : begin with each element as a separate cluster and merge them into successively larger clusters
    - divisive ("top-down") : begin with the whole set and proceed to divide it into successively smaller clusters

- Partitional Algorithms: determine all clusters at once (k-means); require the number of clusters (X-means: automatically determine the number of clusters based on BIC scores; G-means)

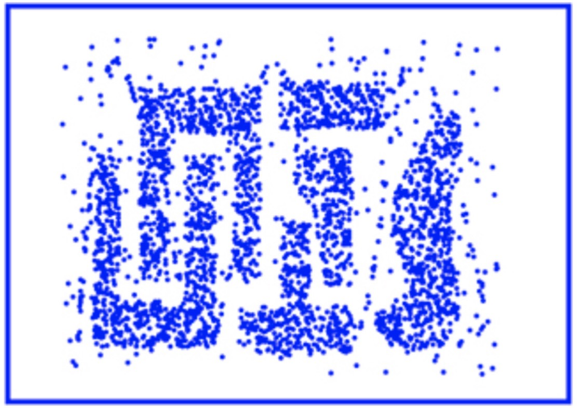
    - Density-based Algorithms: a cluster is regarded as a region in which the density of data objects exceeds a threshold

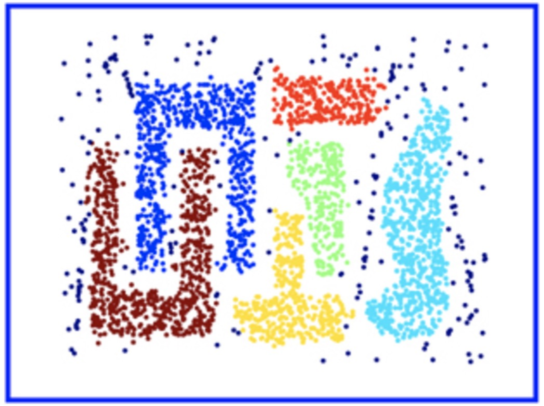
Carnegie Mellon University

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- K-means and HC are suitable for compact and well-separated clusters
- Moreover, they are also severely affected by the presence of noise and outliers in the data

- DBSCAN: Two parameters
    - eps (radius): defines the neighborhood around a data point i.e., if the distance between two points is lower or equal to 'eps' then they are considered as neighbors
    - MinPts: minimum number of neighbors (data points) within eps radius

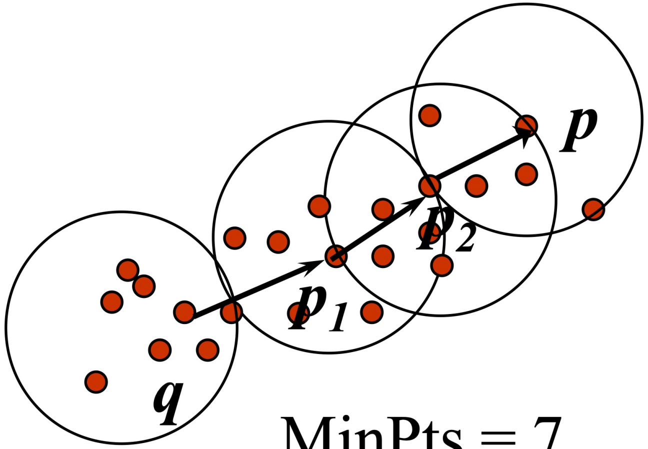




Carnegie Mellon University

DBSCAN: Pseudocode

```
for each $o \in D$ do
    if $o$ is not yet classified then
        if $o$ is a core-object then
            collect all objects density-reachable from $o$
            and assign them to a new cluster.
        else
            assign $o$ to NOISE
```



MinPts = 7

- Core-object: if it has MinPts within Eps

Density Reachable: directly and indirectly

- A point p is directly density-reachable from p2
- p2 is directly density-reachable from p1
- p1 is directly density-reachable from q
- p < p2 < p1 < q form a chain

- p is (indirectly) density-reachable from q
- q is not density-reachable from p

Carnegie Mellon University

DBSCAN: Pseudocode

- Start by defining the ε-neighborhood of point x_n as follows:

$$N_\varepsilon(\mathbf{x}_n) = \{\mathbf{x} \in X | d(\mathbf{x}, \mathbf{x}_n) < \varepsilon\}$$

where N_ε are the data points at a distance smaller than ε from x_n. N_ε, therefore, is a rough estimate of local density. x_n is considered to be a core point if at least minPts are in the neighborhood. A point x_i is said to be density-reachable if it is in the neighborhood of a core-point.

```
• Until all points in $X$ have been visited; do
    • Pick a point $x_i$ that has not been visited.
    • Mark $x_i$ as visited
    • If $x_i$ is a core point, then
        • Find the set of all points, $C$, that are density reachable from $x_i$.
        • $C$ now forms a cluster. Mark all points within that cluster as having been visited.
    • Return the cluster assignments $C_1, \ldots, C_k$, with $k$ the number of clusters. Points that
      have not been assigned to a cluster are considered noise/outliers.
```

Carnegie Mellon University

DBSCAN: Example

- Distance matrix: minimum rotation angle between two (mis)orientations

- Two parameters

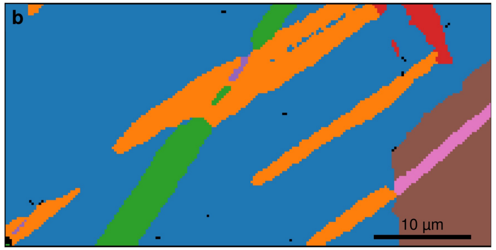
1. ϵ : an angle (in radians 0.05 for both) acting as the upper limit on the distance between two points to be in the neighborhood of one another;

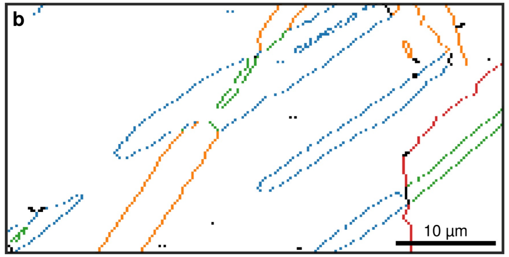
2. n : the minimum number of data points in the core of a cluster (40 for orientations; 10 for misorientations)

Density-based clustering of crystal orientations and misorientations and the orix python library

Duncan N. Johnstone,[a]* Ben H. Martineau,[a] Phillip Crout,[a] Paul A. Midgley[a] and Alexander S. Eggeman[b]

[a]Department of Materials Science and Metallurgy, University of Cambridge, 27 Charles Babbage Road, Cambridge CB3 0FS, United Kingdom, and [b]Department of Materials, The University of Manchester, Oxford Road, Manchester M13 9PL, United Kingdom. Correspondence e-mail: dnj23@cam.ac.uk





Carnegie Mellon University

DBSCAN: Advantages /& Disadvantages

- DBScan does not need to know the number of clusters in the data a priori, as opposed to K-Means.
- DBScan can find arbitrarily shaped clusters. It can even find clusters completely surrounded by (but not connected to) a different cluster. Due to the MinPts parameter, the so-called single-link effect (different clusters being connected by a thin line of points) is reduced.
- DBScan has a notion of noise. Outliers are labeled as Clustering.OUTLIER, which is Integer.MAX_VALUE. (python specific comment)
- DBScan requires just two parameters and is mostly insensitive to the ordering of the points in the database. (Only points sitting on the edge of two different clusters might swap cluster membership if the ordering of the points is changed, and the cluster assignment is unique only up to isomorphism.)

On the other hand, DBScan has the disadvantages of

- In high dimensional space, the data are sparse everywhere because of the curse of dimensionality. Therefore, DBScan doesn't work well on high-dimensional data in general.
- DBScan does not respond well to data sets with varying densities.

Classification of (Hard) Clustering Algorithms

- Hierarchical Algorithms: find successive clusters using previously established clusters
    - agglomerative ("bottom-up") : begin with each element as a separate cluster and merge them into successively larger clusters
    - divisive ("top-down") : begin with the whole set and proceed to divide it into successively smaller clusters

- Partitional Algorithms: determine all clusters at once (k-means); require the number of clusters (X-means: automatically determine the number of clusters based on BIC scores; G-means)

    - Density-based Algorithms: a cluster is regarded as a region in which the density of data objects exceeds a threshold

- For High-D Data: Subspace Methods: clusters that can only be seen in a projection of the data; both objects and features are clustered simultaneously (t-SNE)

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- Means of visualizing the similarities between objects of N-dimensional space in a 2-dimensional scatterplot

- Main advantage of t-SNE is its ability to preserve local structure, which means, roughly, that points which are close to one another in the high-D data set will cluster with one another in the 2-D setting.

- The algorithm models the probability distribution of neighbors around each data point. Here, the term neighbors refers to the set of data points which are closest to the reference data point. In the original, high-dimensional space this is modeled as a Gaussian distribution. In the 2-dimensional output space this is modeled as a t-distribution.

- The goal of t-SNE is to find a mapping onto the 2-dimensional space that minimizes the differences between these two distributions over all points. (minimizes the sum of Kullback-Leibler divergences over all datapoints using GD)

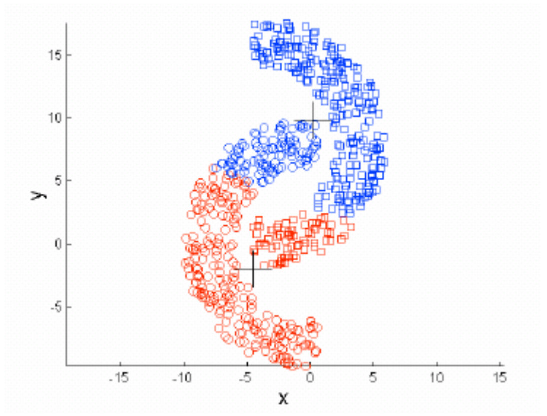
- Within the designed function, the main parameter controlling the fitting is called perplexity. Perplexity is equivalent to the number of nearest neighbors considered when matching the original and fitted distributions for each point.

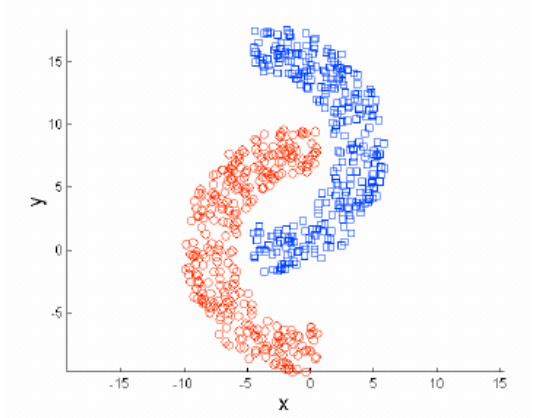
- Hyperparameters: https://distill.pub/2016/misread-tsne/

Carnegie Mellon University

Clustering Challenges

- Data pre-processing
- Define appropriate dissimilarity
- Selecting appropriate method
  - What do you know about the data?
  - What methods have been successfully used on similar data?
  - Try multiple methods and visualize results
- Number of clusters
  - Heuristic, e.g., Gap statistic
- Visualize Results!
  - Most important check is to have an expert look at the results





Questions