2695 Introduction to Machine Learning
Masters Program in Economics, Finance and Management

NOVA SCHOOL OF BUSINESS & ECONOMICS

6a Video Lecture

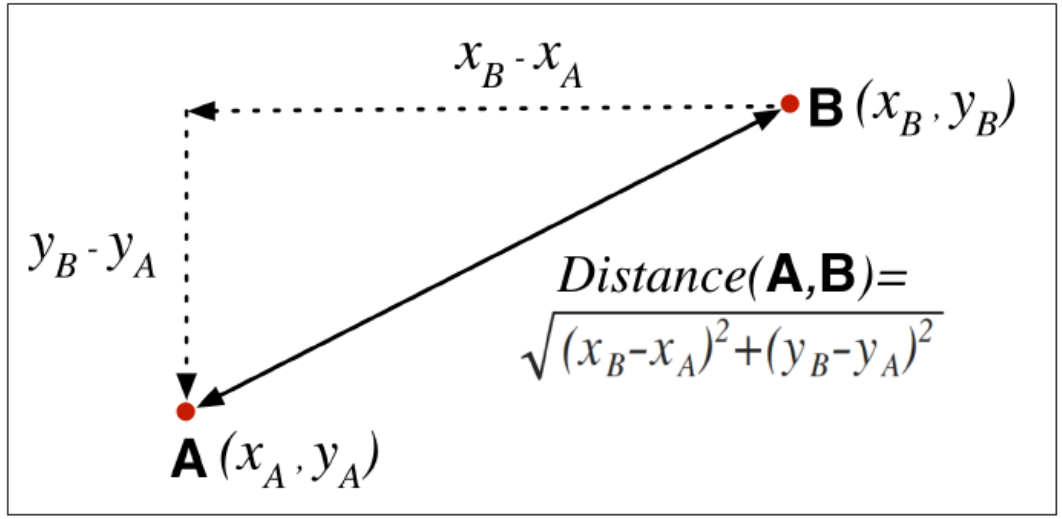
SIMILARITY, DISTANCE AND NEAREST NEIGHBORS

Euclidean distance

Data points as feature vectors

Attribute	Person A	Person B
Age	23	40
Years at current address	2	10

Coordinates are just the values of the two features of the objects



$$d(A, B) = \sqrt{(23 - 40)^2 + (2 - 10)^2}$$

Euclidean distance in n dimensions

$$\sqrt{(d_{1,A} - d_{1,B})^2 + (d_{2,A} - d_{2,B})^2 + \cdots + (d_{n,A} - d_{n,B})^2}$$

Distance Metrics

Manhattan Distance (L1 norm)

- Distance measure between data points in a grid
- Less sensitive to outliers and thus more suitable for high-dimensional data

$$d_{\text{Manhattan}}(\mathbf{X},\mathbf{Y}) = \| \mathbf{X} - \mathbf{Y} \|_1 = | x_1 - y_1 | + | x_2 - y_2 | + \cdots$$

Jaccard distance

- Given two objects, X and Y, the Jaccard distance is the proportion of all the characteristics that are shared by the two objects.
- Used in recommender systems

$$d_{\text{Jaccard}}(X, Y) = 1 - \frac{| X \cap Y |}{| X \cup Y |}$$

Cosine distance

- Used in recommender systems and text similarity

$$d_{cosine}(\mathbf{X},\mathbf{Y}) = 1 - \frac{\mathbf{X} \cdot \mathbf{Y}}{\| \mathbf{X} \|_2 \cdot \| \mathbf{Y} \|_2}$$

Nearest neighbors for Classification (KNN)

Example: credit card marketing problem:
Predict **whether a new customer will respond to a credit card** offer based on how other, similar customers have responded.

Customer	Age	Income (1000s)	Cards	Response (target)	Distance from David
David	*37*	*50*	*2*	*?*	0
John	35	35	3	Yes	$\sqrt{(35-37)^2+(35-50)^2+(3-2)^2} = 15.16$
Rachael	22	50	2	No	$\sqrt{(22-37)^2+(50-50)^2+(2-2)^2} = 15$
Ruth	63	200	1	No	$\sqrt{(63-37)^2+(200-50)^2+(1-2)^2} = 152.23$
Jefferson	59	170	1	No	$\sqrt{(59-37)^2+(170-50)^2+(1-2)^2} = 122$
Norah	25	40	4	Yes	$\sqrt{(25-37)^2+(40-50)^2+(4-2)^2} = 15.74$

Example from: Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking, F. Provost, T. Fawcett

Nearest neighbors for Probability Estimation

Example: credit card marketing problem:
The goal is to **estimate the probability of a new customer** responding to a credit card offer based on how other, similar customers have responded.

Customer	Age	Income (1000s)	Cards	Response (target)	Distance from David
David	*37*	*50*	*2*	*?*	0
John	35	35	3	Yes	$\sqrt{(35-37)^2+(35-50)^2+(3-2)^2}=15.16$
Rachael	22	50	2	No	$\sqrt{(22-37)^2+(50-50)^2+(2-2)^2}=15$
Ruth	63	200	1	No	$\sqrt{(63-37)^2+(200-50)^2+(1-2)^2}=152.23$
Jefferson	59	170	1	No	$\sqrt{(59-37)^2+(170-50)^2+(1-2)^2}=122$
Norah	25	40	4	Yes	$\sqrt{(25-37)^2+(40-50)^2+(4-2)^2}=15.74$

- If we score for the Yes class, so that Yes=1 and No=0, we can average these into a score of *2/3* for David.

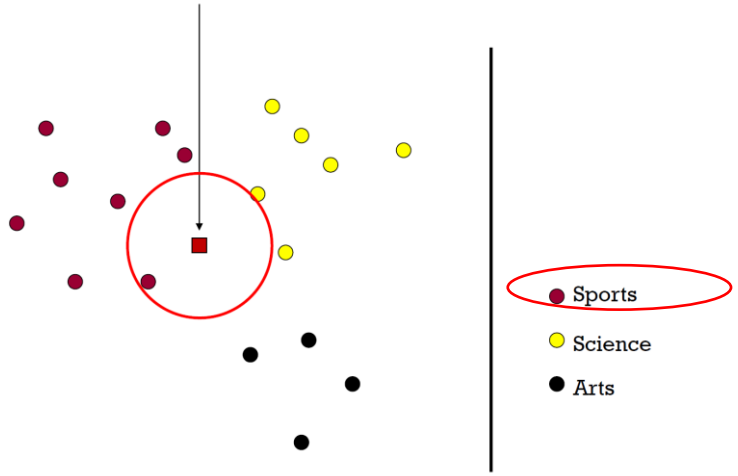
Nearest neighbors for Regression

Example: credit card marketing problem:
The goal is **to estimate the income of a new customer**, based on incomes of other, similar customers.

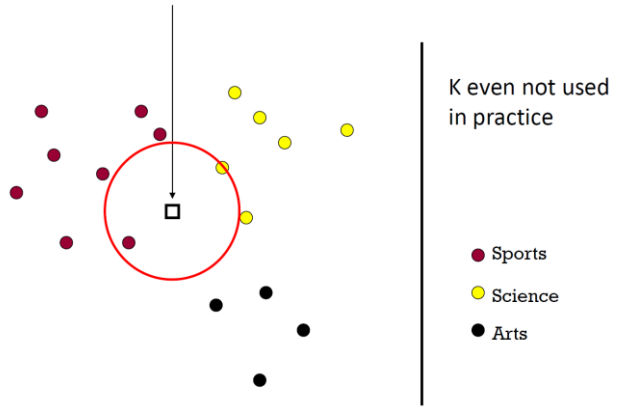
Customer	Age	Income (1000s)	Cards
David	*37*	**?**	*2*
John	35	35	3
Rachael	22	50	2
Ruth	63	200	1
Jefferson	59	170	1
Norah	25	40	4

How many neighbors should we consider?

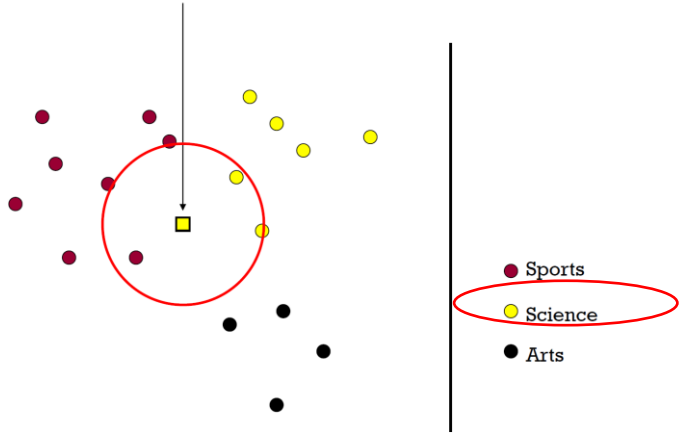
1-Nearest Neighbor (kNN) classifier



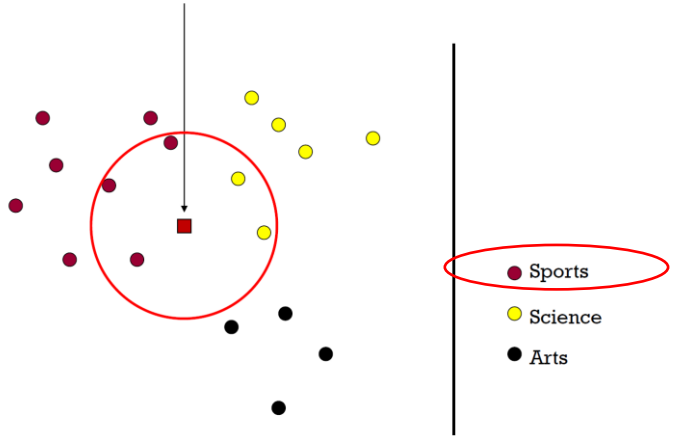
2-Nearest Neighbor (kNN) classifier



3-Nearest Neighbor (kNN) classifier

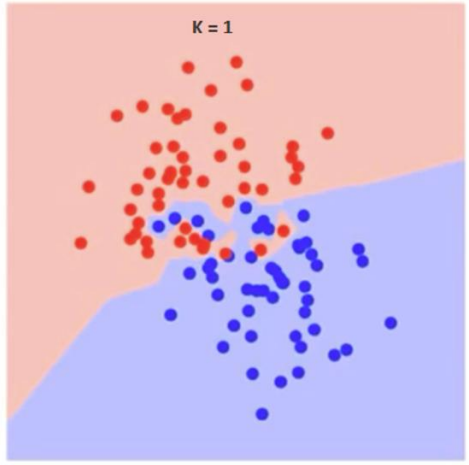


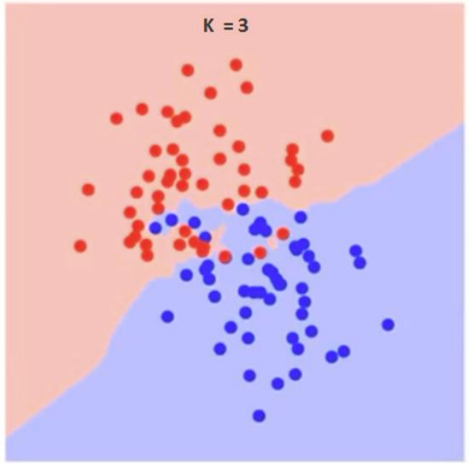
5-Nearest Neighbor (kNN) classifier



Impact of K on the decision boundary

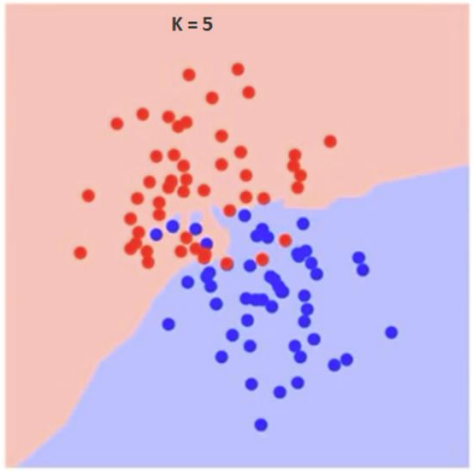
K=1

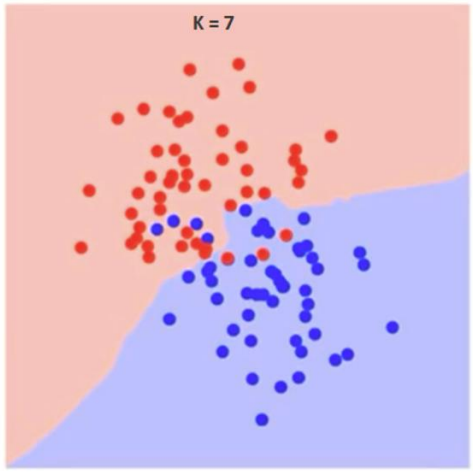




K=3

K=5





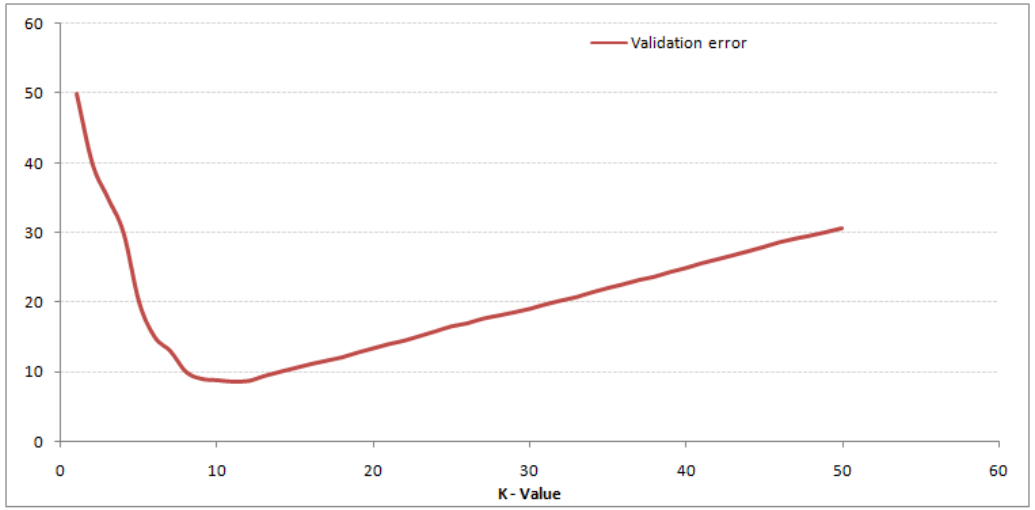
K=7

How to choose K in KNN?

- With K=N (number of instances) we get a simple model (**underfitting**).
  - Classification: the majority class in the entire dataset
  - Probability estimation: the “base rate” probability (percentage of majority class in the population)
  - Regression: the average of all the target values

- With K=1, we get an extremely complex model (**overfitting**).

- **K is a hyperparameter**, it can be chosen using hyperparameter tuning with cross validation.



How to combine neighbors target values?

Classification

- **Majority voting**
    - Pick the class of the majority of neighbors
    - Get an odd number of neighbors to break ties
    - Ignore how close is each neighbor to the new instance

- **Weighted voting (similarity moderated voting)**
    - Each neighbor's contribution is scaled by its similarity
    - Weights: inverse of the square of the distance
    - Contributions: proportional to the weights, but adding up to one
    - Weighted scoring has a nice consequence in that it reduces the importance of deciding how many neighbors to use.

Measuring similarity (more detailed look)

- Choose a distance metric
- Encode categorical attributes into numerical
- **Scale the features**

Attribute	Person A	Person B
Age	23	40
Years at current address	2	10
Residential status (1=Owner, 2=Renter, 3=Other)	2	1
Income	50,000	90,000

Age might have a range from 18 to 100, while Income might have a range from $1000 to $1,000,000.
Without scaling, our distance metric would consider ten dollars of income difference to be as significant as ten years of age difference

KNN summary

- KNN algorithm is one of the simplest algorithms.
- Used for classification, regression, probability estimation, imputing missing values
- Training: storing the instances (fast)
- Prediction: comparing a new data point with the training data points (computationally intensive).
- It is a non-parametric method, instance-based learning (lazy learning):
  No explicit model, prediction is done by comparing a new data point with the training data points.
- The accuracy of the algorithm can be severely degraded by the presence of noisy or irrelevant features.

2695 Introduction to Machine Learning
Masters Program in Economics, Finance and Management

NOVA SCHOOL OF BUSINESS & ECONOMICS

6b Video Lecture

CLUSTERING

Clustering

- Another application of similarity:
  - objects within groups are similar
  - objects in different groups are not so similar

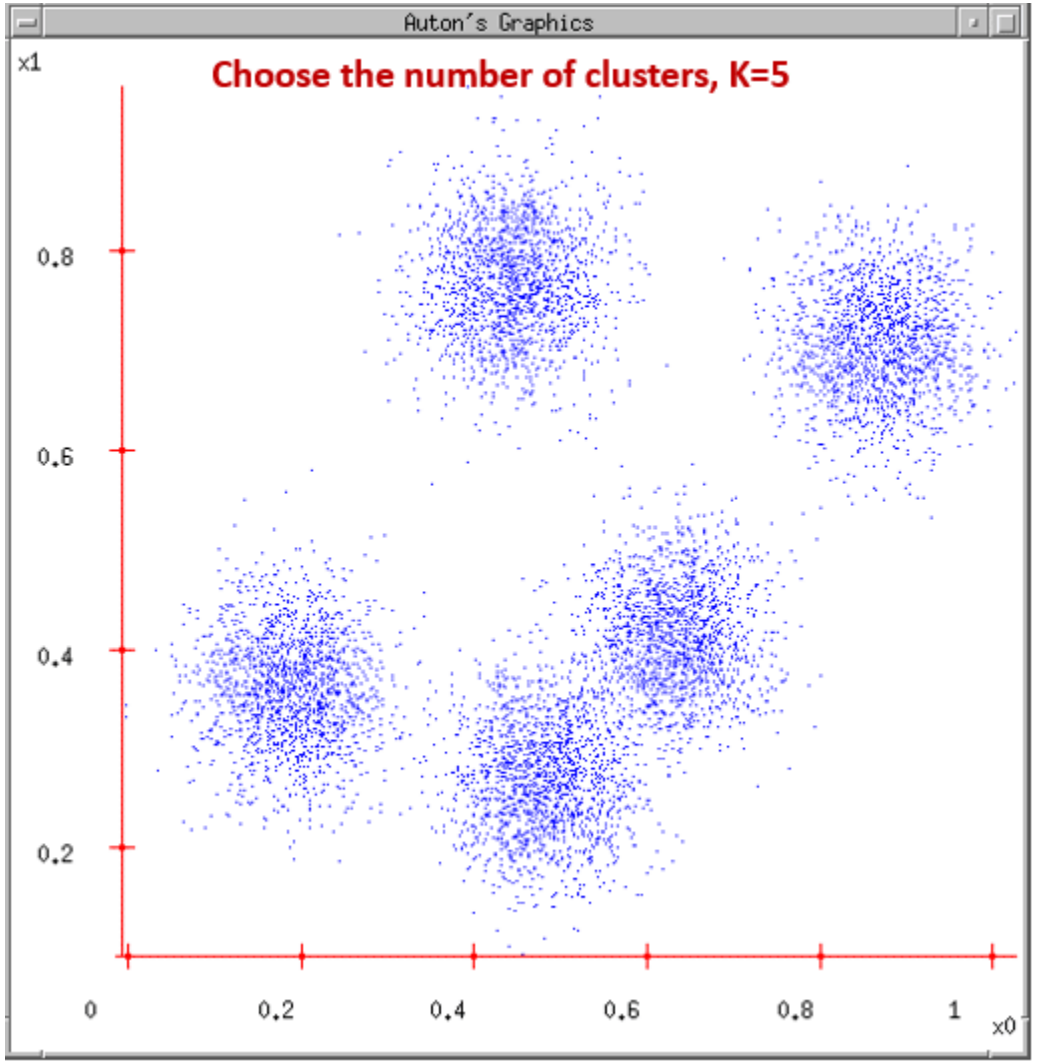
- Groups individuals in a population together by similarity:
  - not driven by a specific purpose
  - no labels in the data and no class values denoting a priori grouping of the data instances

- Applications
  - market segmentation
  - social network analysis
  - genomics

- Methods
  - Partitioning methods (Centroid-based clustering)
  - Density-based methods

K-Means example

1. Choose a number of clusters *(e.g., K=5)*
2. K cluster center positions chosen at random
3. Each datapoint assigned to the closest centroid
4. Find the mean of all the points in the cluster
5. …and Center jumps there
6. …Repeat until terminated!



K-means algorithm

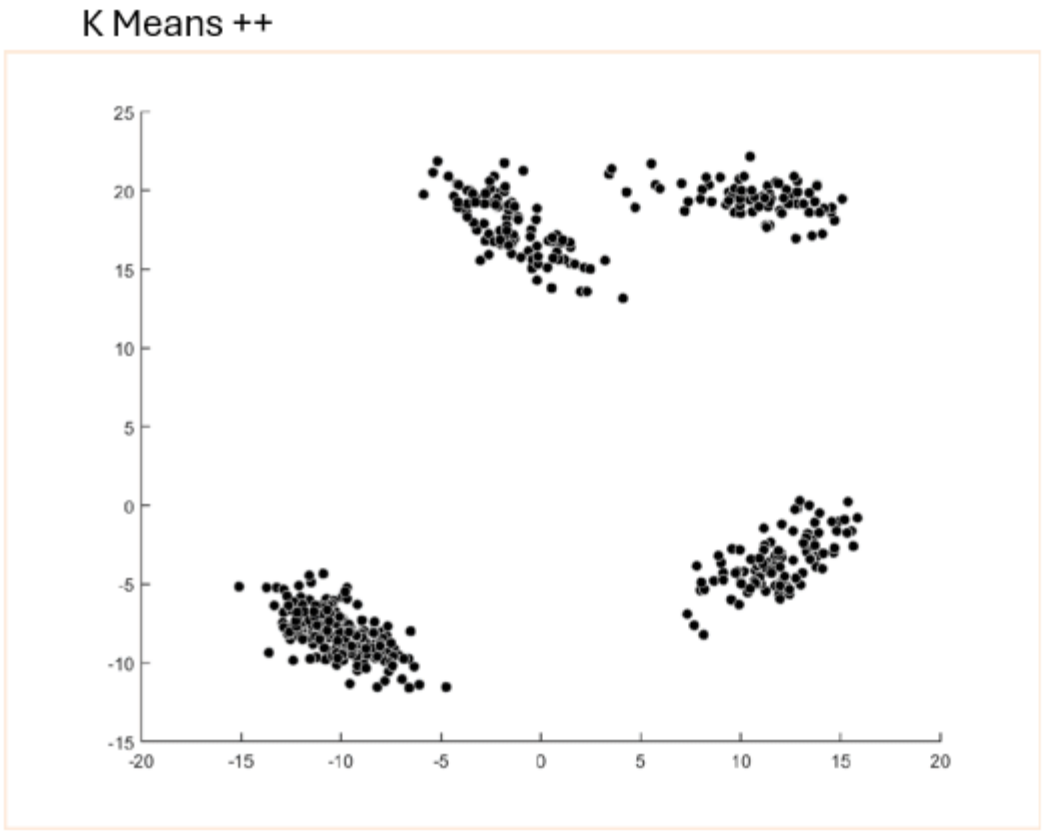
- Performs two stages:
  - **Cluster assignment step**: Assign each data point to the nearest cluster, such that it has the smallest squared Euclidean distance to its centroid.
  - **Update centroid step:** Calculate the means of the features in the new clusters to be the new centroids.
- K-means algorithm aims to choose centroids that minimize the **inertia**, or within-cluster sum of squares criterion:

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

- Guaranteed to converge when using Euclidean distance.
- For every different run of the algorithm on the same dataset, the initial centers might be different. This may lead to different clustering results.
- No guarantee that a single run will result in a good clustering (local, not global optimum):
  - **Repeat K-means:** Repeat the algorithm and starting with different random centroids each time, and pick the clustering approach that has the lowest **inertia**.
  - **K-Means++:** smart centroid initialization technique.

K-Means++

1. Select the first centroid randomly from the dataset
2. Compute distances of all other points from this centroid
3. Select the next centroid such that points that are farther from existing centroids have a higher probability of being chosen as the next centroid.
4. Repeat steps 2-3 until K centroids are chosen.
5. Proceed with regular K-Means clustering using these centroids.



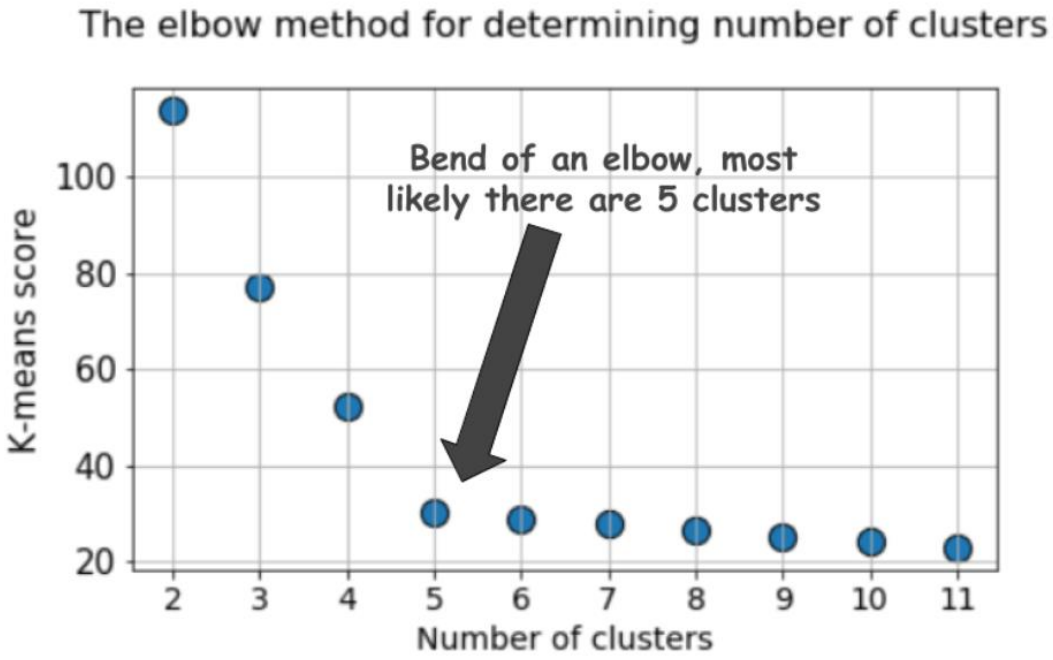
How to choose the value of K in K-means?

- There is no method for determining the exact value of K.
- Experiment with different *K* values and see which ones generate good results.
- Challenge in unsupervised learning- evaluating whether the algorithm learned something useful:
  - Clustering results analyzed by domain experts to determine whether the clusters make sense.
  - Objective measures can be used and calculated for increasing values of *K.*

Elbow Method

- Calculate the sum of the squared distances between each data point and its corresponding centroid (inertia)
  - Test for different K
  - Sum falls rapidly until right K, then falls much more slowly
  - It might be hard to determine where the elbow of the curve is actually

$$\sum_{i=1}^{N}(x_i - C_k)^2$$



Unfortunately, the elbow method often does not work well in practice because there may not be a well-defined elbow.

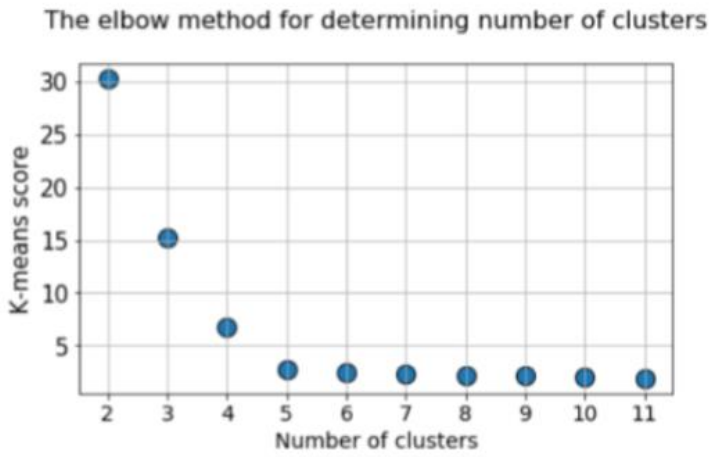
Silhouette method

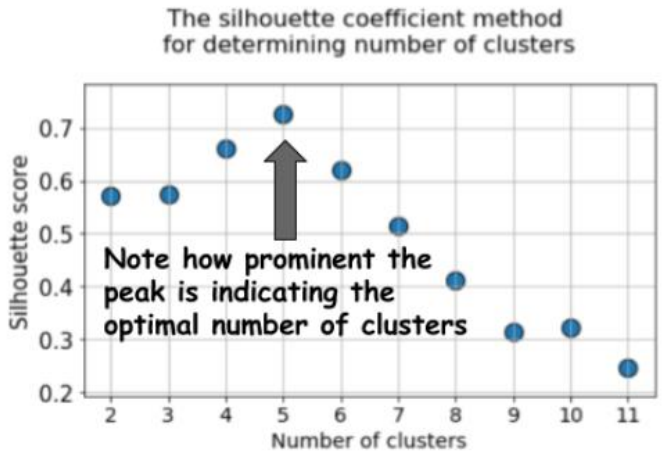
Measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

1. Calculate the **cluster cohesion**, a^i as the average distance between an example, i and all other points in the same cluster.
2. Calculate the **cluster separation**, b^i, from the next closest cluster as the average distance between the example, i, and all examples in the nearest cluster.
3. Calculate the silhouette, s^i, as the difference between cluster cohesion and separation divided by the greater of the two:

$$s^{(i)} = \frac{b^{(i)} - a^{(i)}}{max\{b^{(i)}, a^{(i)}\}}$$

4. The silhouette coefficient is then the average of the silhouette coefficients across all the data point: bounded between -1 and 1





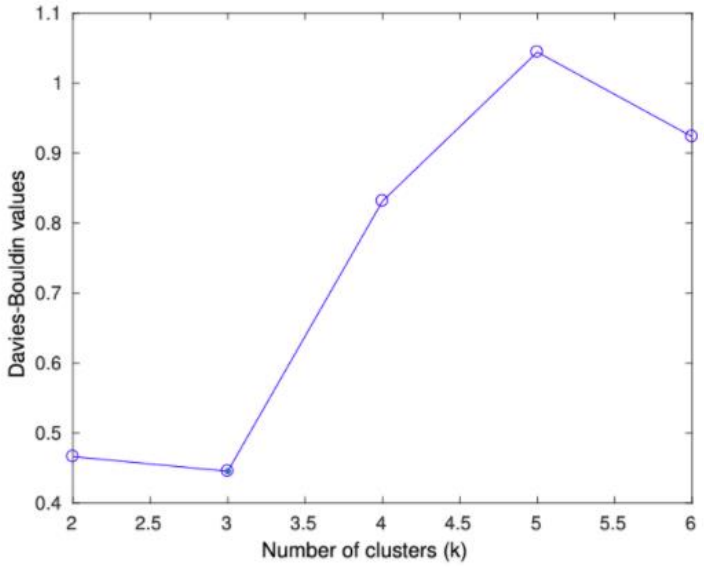
Davies-Bouldin index

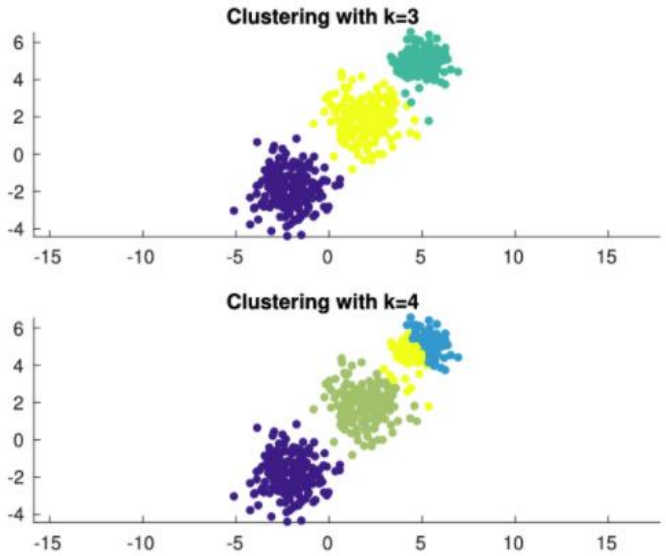
- Based on a ratio between “within-cluster” and “between-cluster” distances.
- We should choose the clustering with the smallest Davies–Bouldin index value.

In practice plot all three metrics (elbow, Silhouette, DBI):

- Look for the Elbow Point in the inertia plot
- Pick K with high Silhouette Score
- Pick K with low Davies-Bouldin Index

If results are inconsistent, **balance the three metrics** to select the most reasonable K.





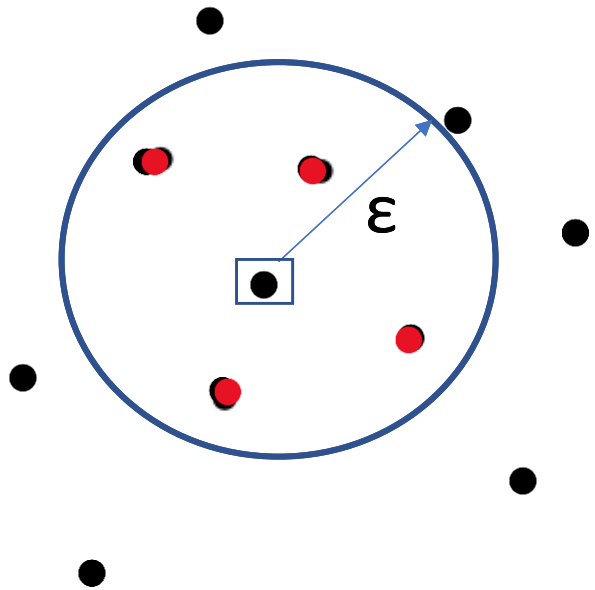
DBSCAN: Density-based clustering algorithm

Density based clustering: clusters with arbitrary shapes as it is not using the distance between data points for clustering

DBSCAN has two hyperparameters:

- Epsilon (ε): distance we use to decide whether another point is a neighbor.
- Minimum number of neighbors: number of neighbors needed to say a region is **dense:**

  if a point has at least minNeighbors, it is called a **core** point.



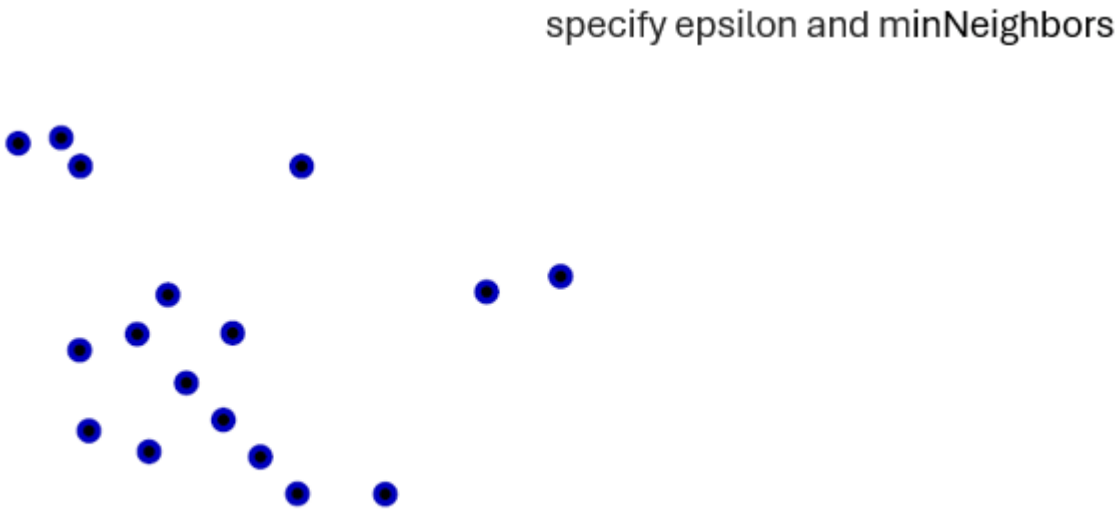
Red points are neighbors

If minimum number of neighbors was set to 3, then this point would be a core point

DBSCAN algorithm: example

DBSCAN algorithm (simplified)

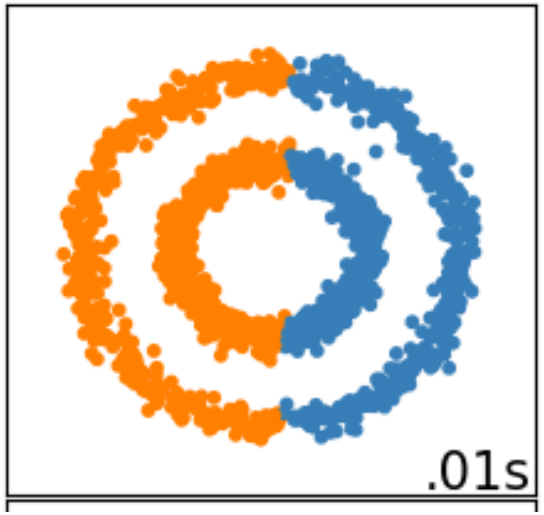
- check the neighborhood of each point
- find dense neighborhoods
- merge dense neighborhoods which are connected



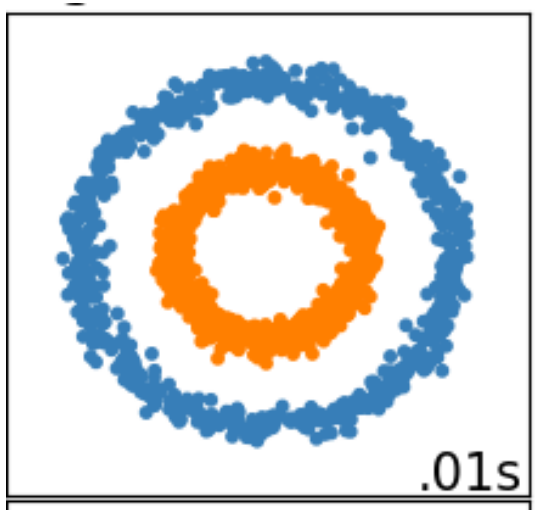
DBSCAN properties

- Does not try to partition space.
- Clusters are defined by dense regions.
- Examples in non-dense regions don’t get clustered.
  Good or bad, depending on the application (anomaly detection).
- Two parameters: ε and minNeighbors

KMEANS



DBSCAN



Methods for describing clusters

- Compute summary statistics for each cluster
- Use Decision Trees for cluster differentiation (what differentiates one cluster from others)
  - Use cluster assignments to label examples (for example, cluster 0 is class 0)
  - Do classification with decision tree. For *k* clusters :
    - set up a *k*-class multiclass classification
    - set up *k* classifiers, each trying to differentiate one cluster from all the other *(k–1)* clusters.
  - Get rules from the tree
- Visualize clusters

NOVA SCHOOL OF BUSINESS & ECONOMICS

6c Video Lecture

DIMENSIONALITY REDUCTION

Dimensionality reduction for visualization

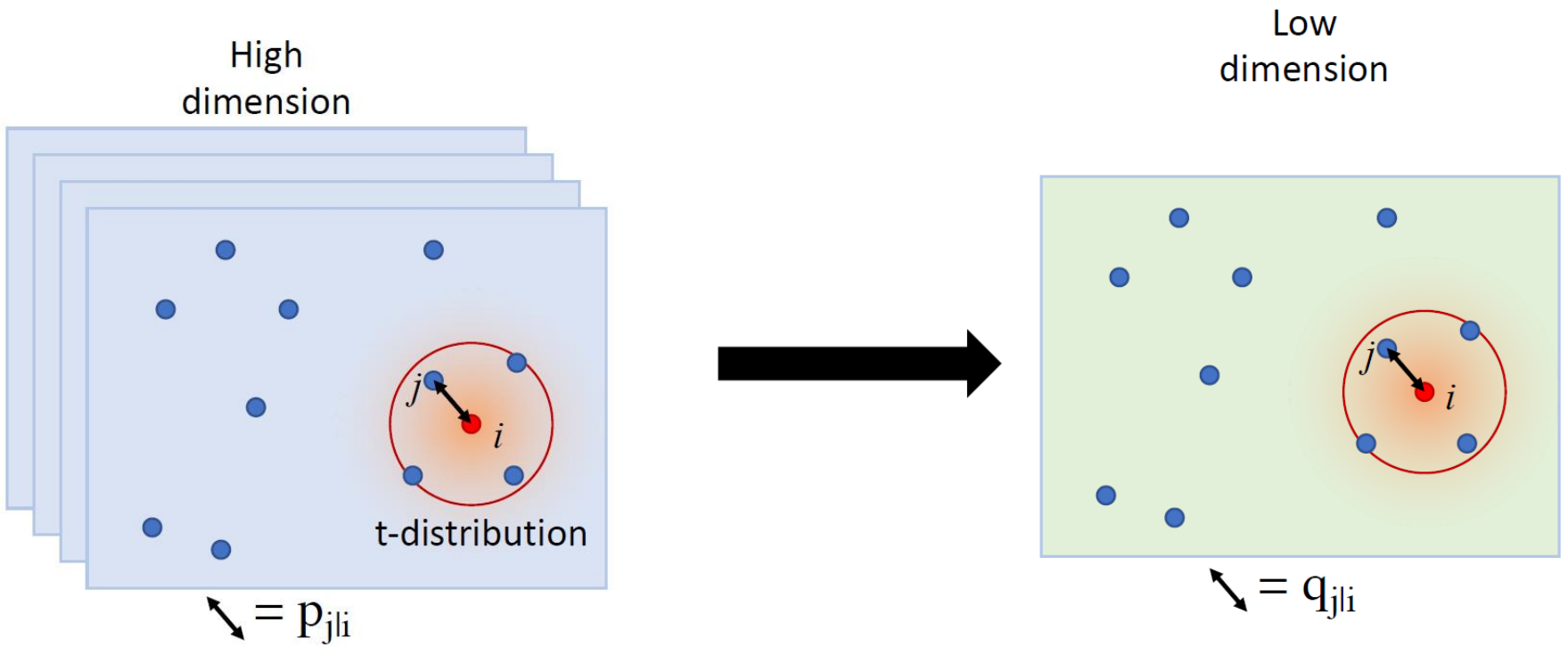
Dimensionality reduction methods transform the data in a high-dimensional space, to a space with fewer dimensions

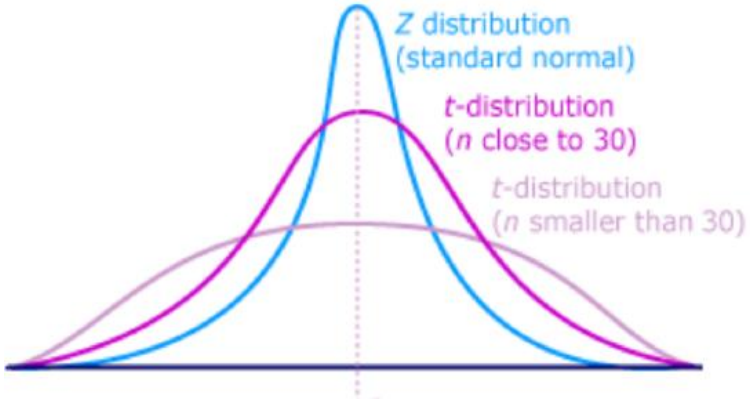
- High-dimensional data is difficult to interpret
- Data often contains redundant or irrelevant features
- Helps in detecting patterns, clusters, and anomalies

Approaches based on manifold learning:

- t-distributed stochastic neighbor embedding (t-SNE)
- Uniform Manifold Approximation and Projection (UMAP)

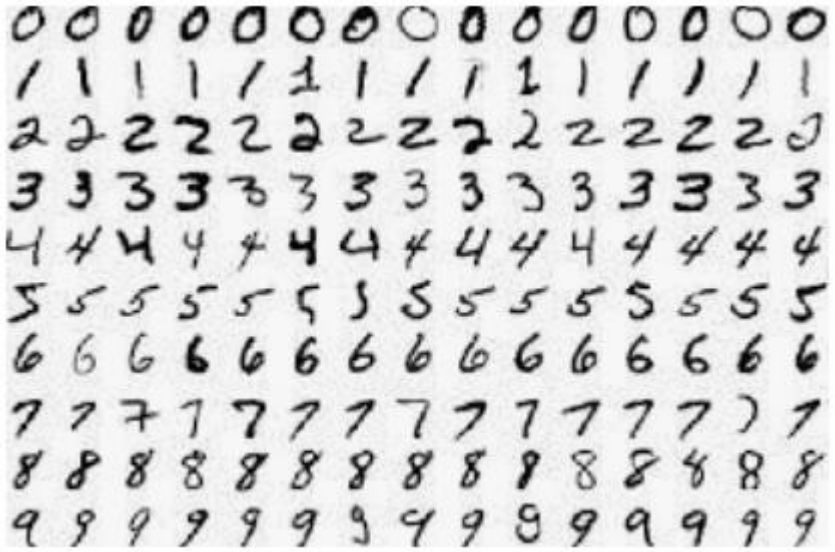
t-Distributed Stochastic Neighbor Embedding



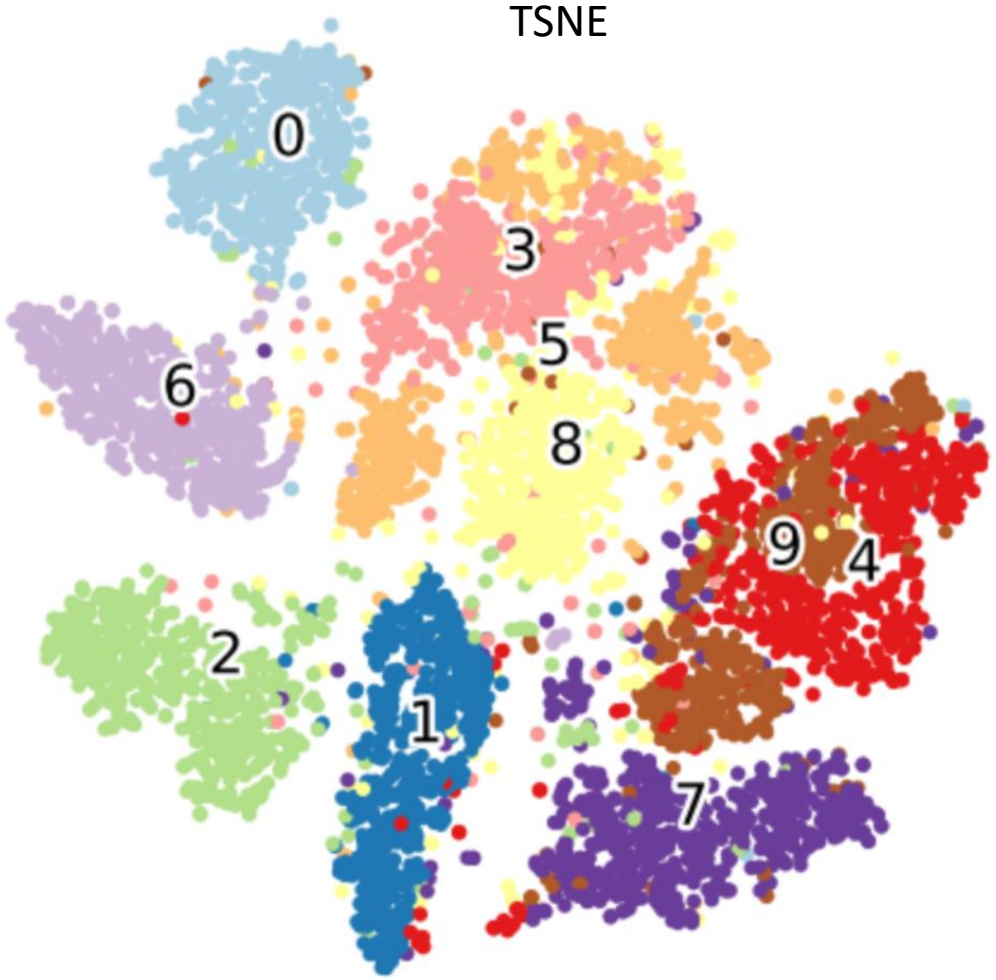


t-SNE for visualizing a dataset of handwritten digits (MNIST)

MNIST dataset



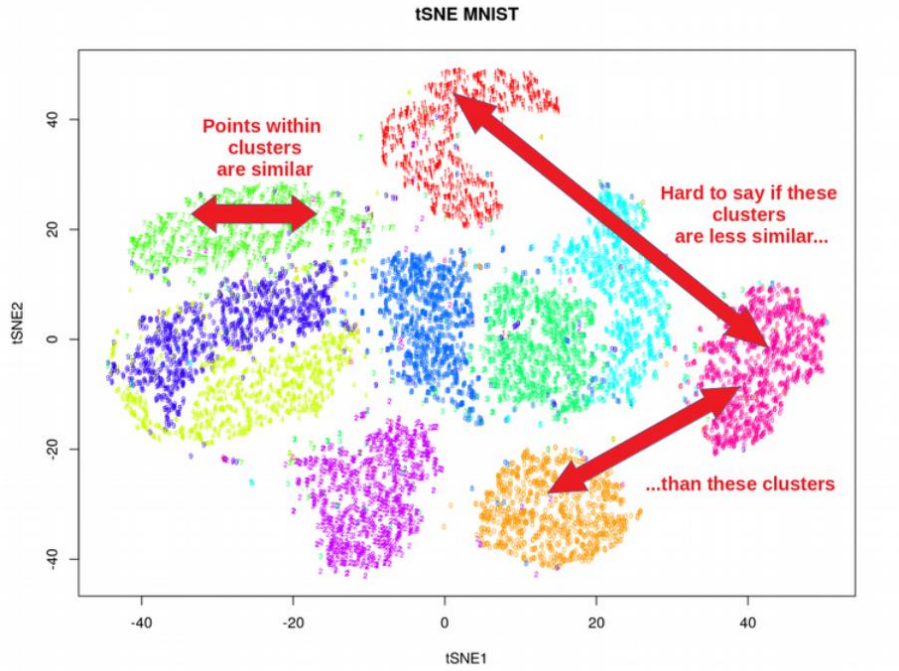
TSNE



https://towardsdatascience.com/t-distributed-stochastic-neighbor-embedding-t-sne-bb60ff109561
https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

t-SNE characteristics

- Its cost function is not convex , optimal t-SNE is not guaranteed to be computed.
- It does not learn an explicit function to map new points, cannot be applied to a new data set
- It is a stochastic algorithm, so it will never produce the same output, unless a random seed is fixed.
- It has a couple of hyperparameters that need to be defined empirically:
  Perplexity: balances between local and global aspects of data
- It does not preserve a global data structure, only local, and the distances within a group are slightly meaningful, but not between groups.



Uniform Manifold Approximation and Projection (UMAP)

- UMAP constructs a high dimensional graph representation of the data then optimizes a low-dimensional graph to be as similar as possible.

- Compared to t-SNE:
  - It preserves more of the global structure.
  - It has lower run time performance.
  - It has no computational restrictions on embedding dimension, making it more a general-purpose dimensionality reduction technique.

- It is still stochastic algorithm, different runs with the same hyperparameters can yield different results.

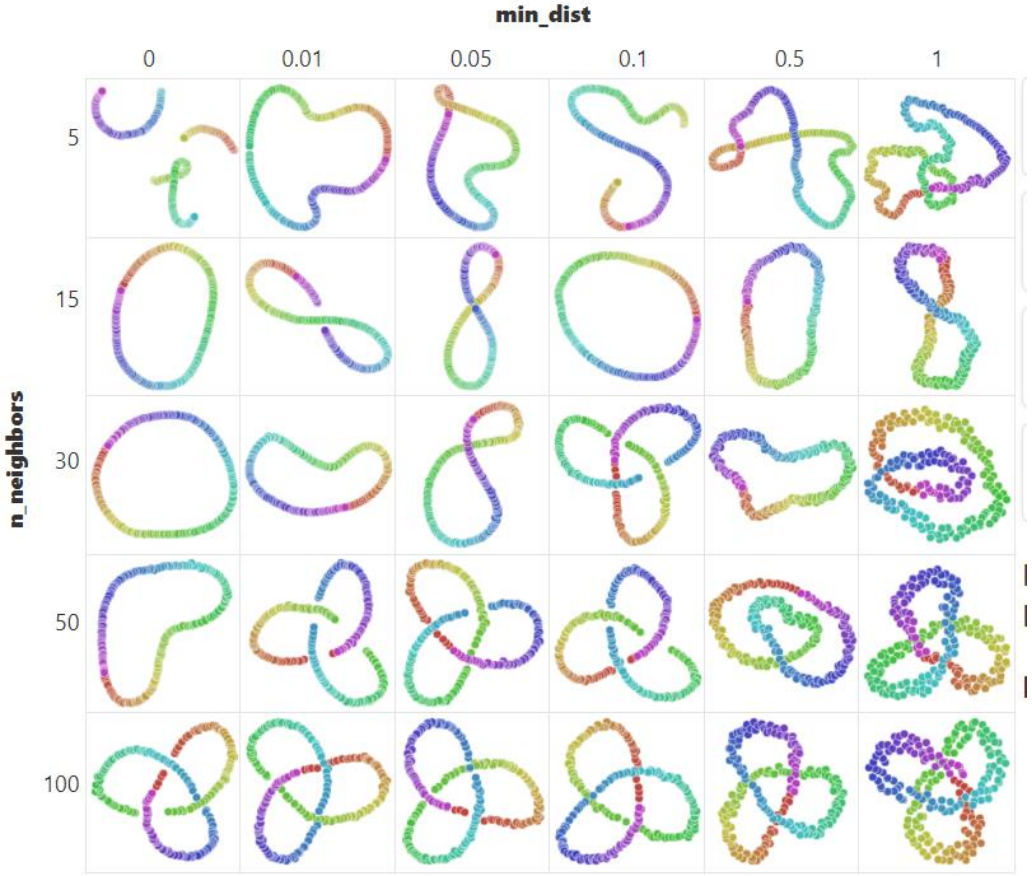
- Global positions of clusters are better preserved than in TSNE, but the distances between clusters might not be meaningful.

- It is sensitive to the choice of the hyperparameters.

UMAP hyperparameters

Balance between local and global structure controlled with parameters:

- *number of neighbors*: low values more focus on local structure, high values focus on the big-picture
- *minimum distance:* low values more tightly packed points



https://pair-code.github.io/understanding-umap/