CLUSTERING

MOHAMMAD GHODDOSI

WHAT IS CLUSTERING

- grouping a set of objects
- objects in the same group are more similar to each other than to those in other groups.

CLUSTERING VS CLASSIFICATION

- Clustering is unsupervised
  - We just have input
  - We don’t know possible outputs
- Classification is supervised
  - We have input and label
  - We know possible outputs

K-MEANS

Input:

D= {t1, t2, …. Tn } // Set of elements

K // Number of desired clusters

Output:

K // Set of clusters

K-Means algorithm:

Assign initial values for $m1, m2, \ldots. mk$

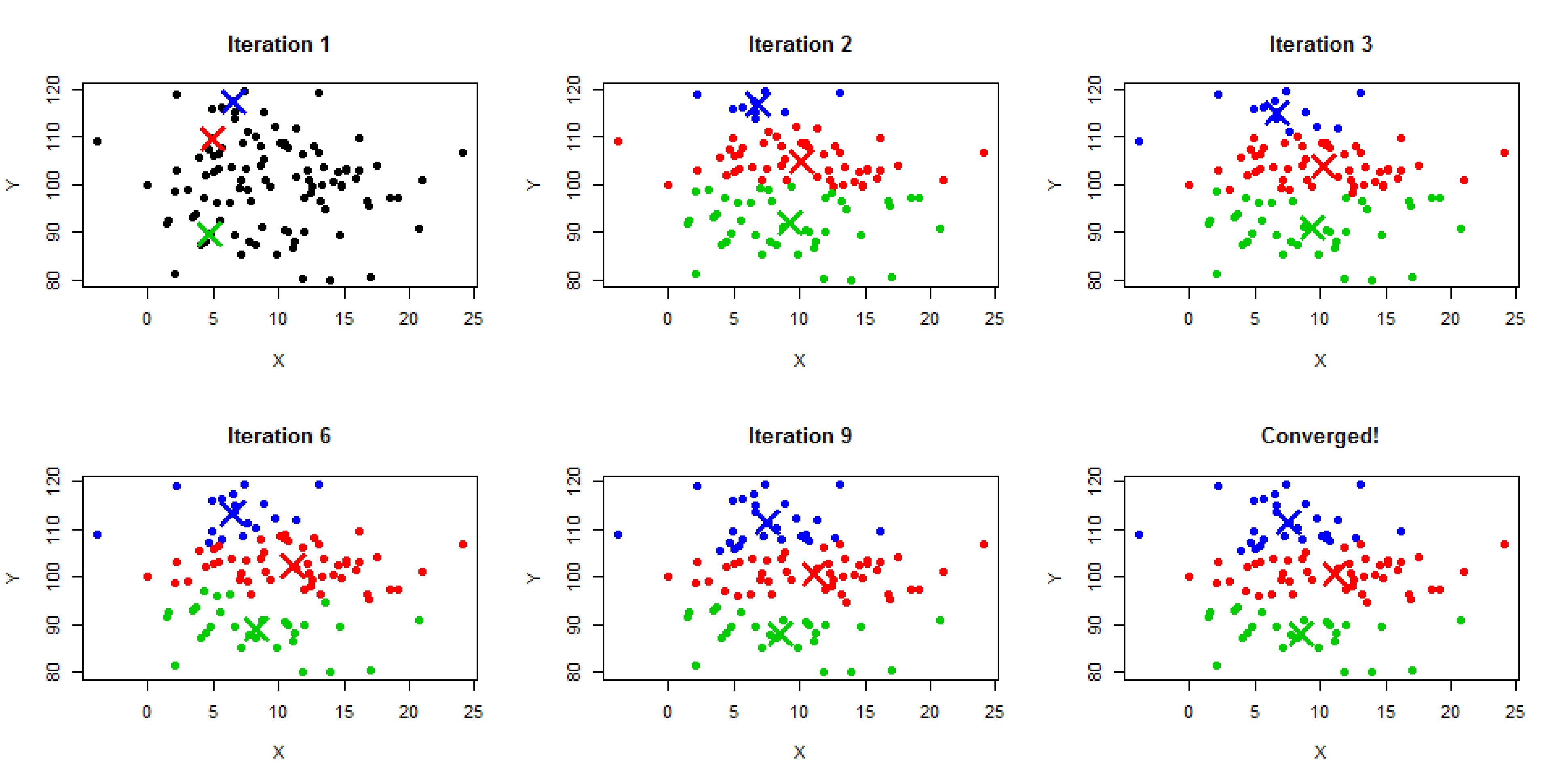
repeat

assign each item t_i to the clusters which has the closest mean;

calculate new mean for each cluster;

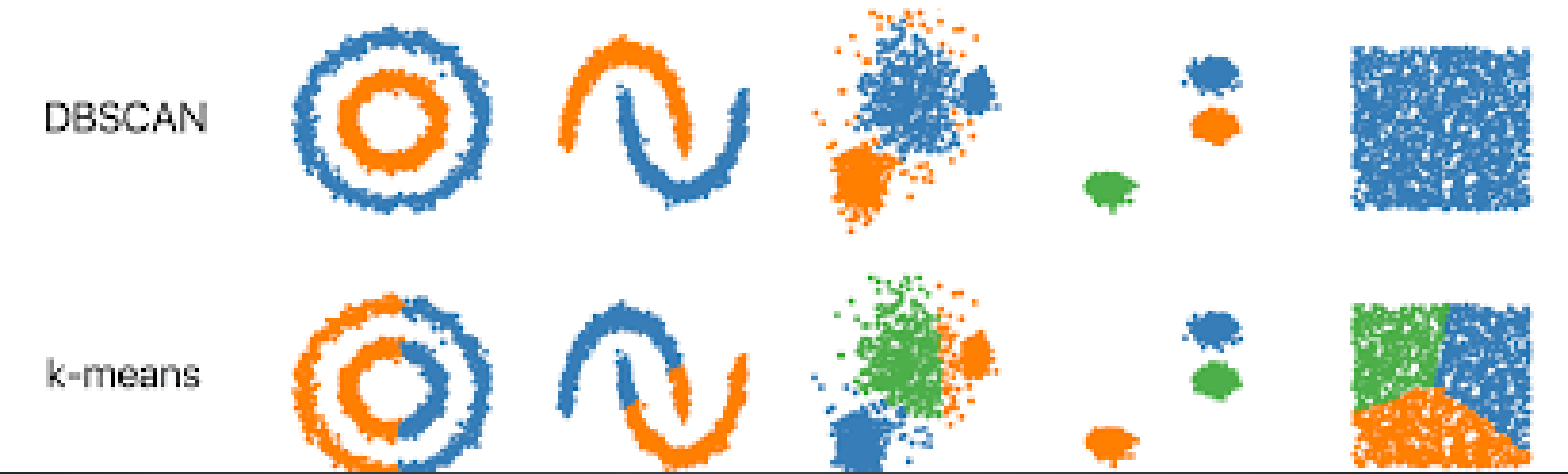
until convergence criteria is met;

K-MEANS

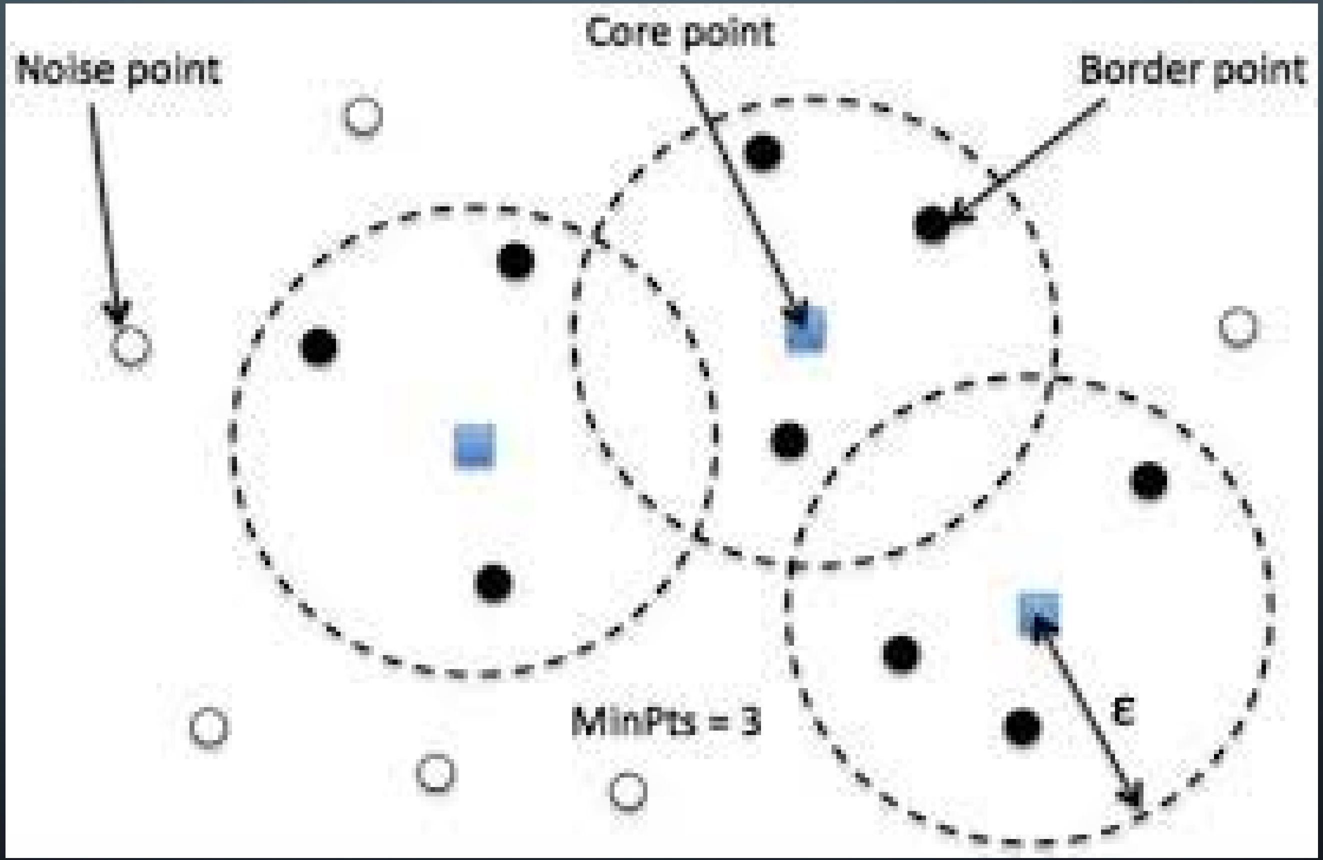


Iteration 1
Iteration 2
Iteration 3
Iteration 6
Iteration 9
Converged!
Y
X
120
110
100
90
80
0
5
10
15
20
25

DB-SCAN



CORE, BORDER AND NOISE POINTS

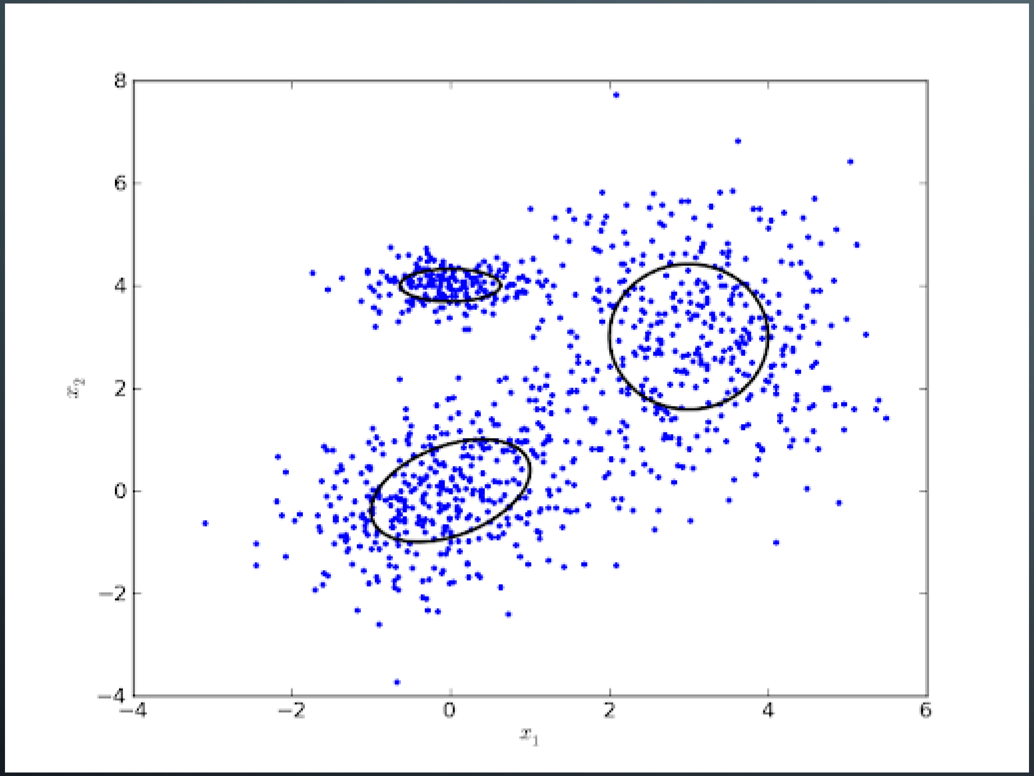


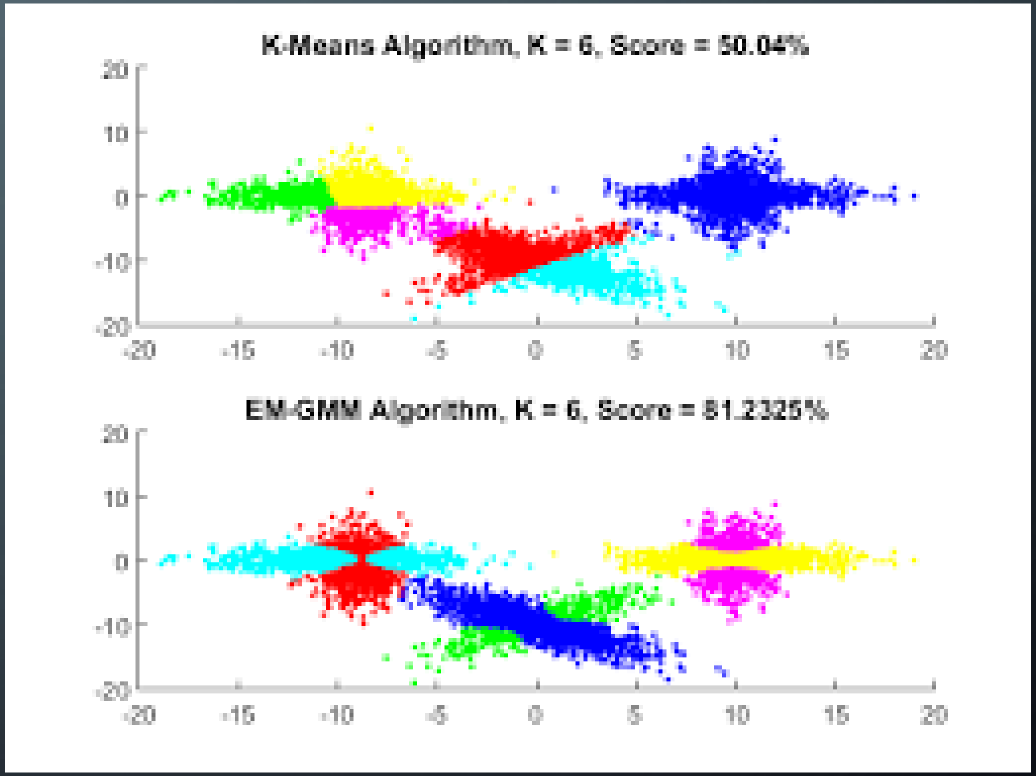
DB-SCAN

Algorithm 1. DBSCAN algorithm

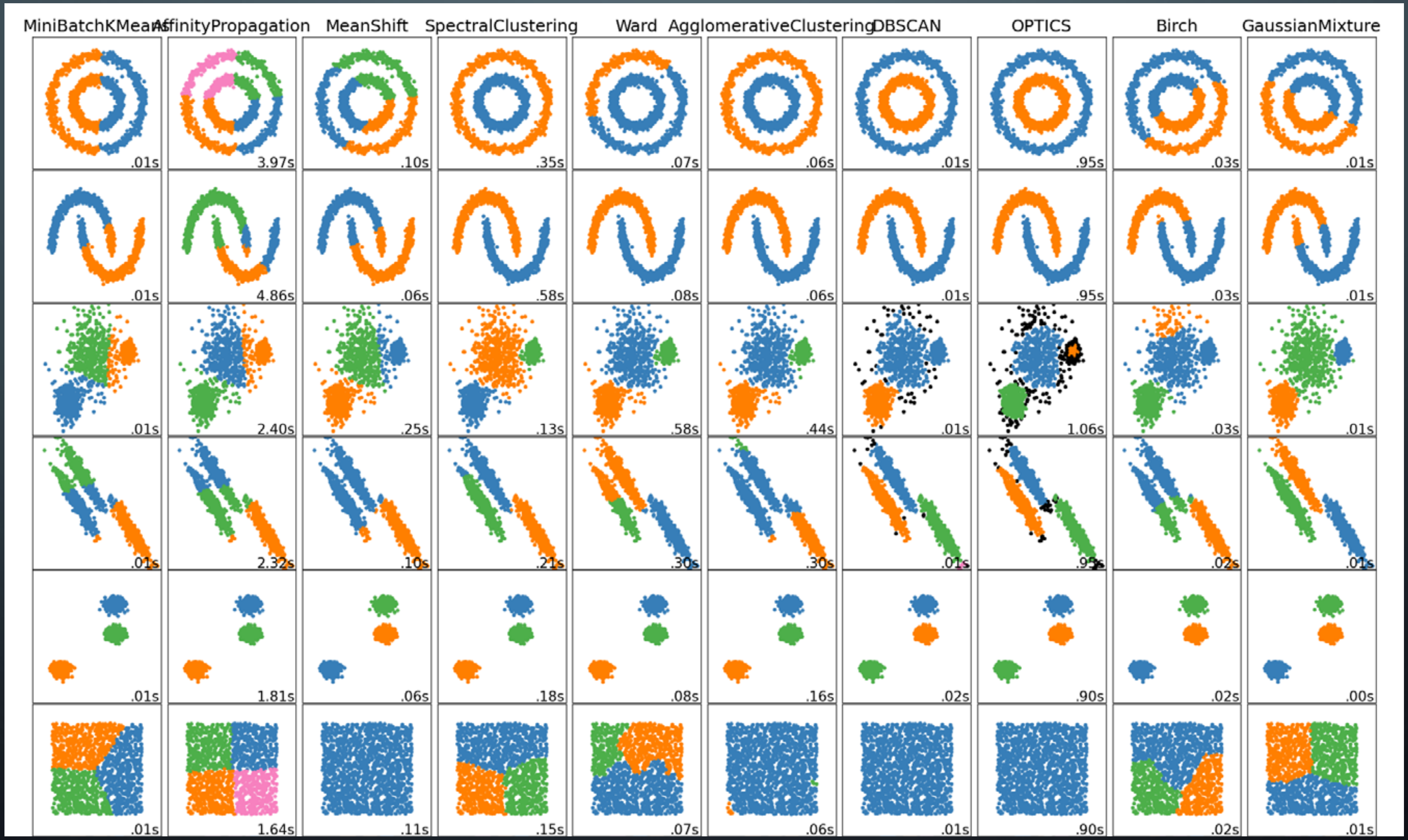
```
Data:
Dataset - D,
distance - ε,
minimum number of points to create dense region - minPts
begin
    C ⟵ 0
    for each point P in dataset D do
        if P is visited then
            Continue to next P
        end
        else
            mark P as visited
            nbrPts ⟵ points in ε-neighborhood of P (distance function)
            if sizeof(nbrPts) < minPts then
                mark P as NOISE
            end
            else
                C ⟵ NewCluster
                Call Expand Cluster Function(P, nbrPts, C, minPts)
            end
        end
    end
end
```

GAUSSIAN MIXTURE MODEL





CLUSTERING IN DIFFERENT DATA



jupyter

DIMENSIONALITY REDUCTION

MOHAMMAD GHODDOSI

DIMENSIONALITY REDUCTION

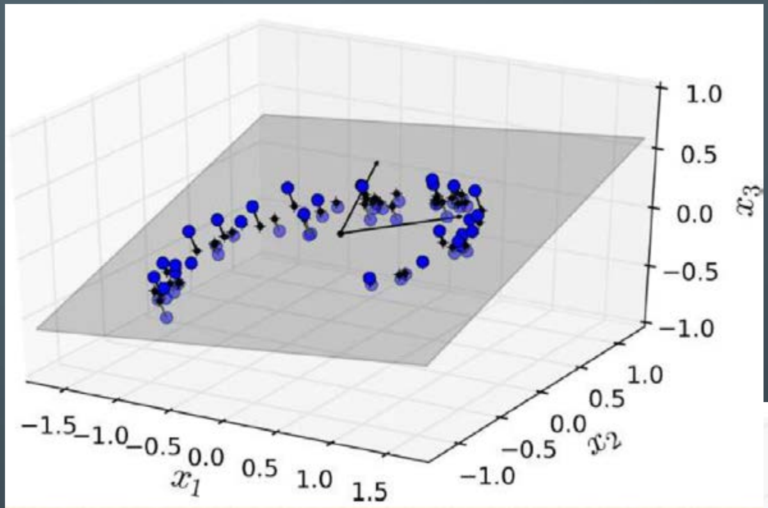
- transformation of data from a high-dimensional space into a low-dimensional space
- low-dimensional representation retains some meaningful properties of the original data

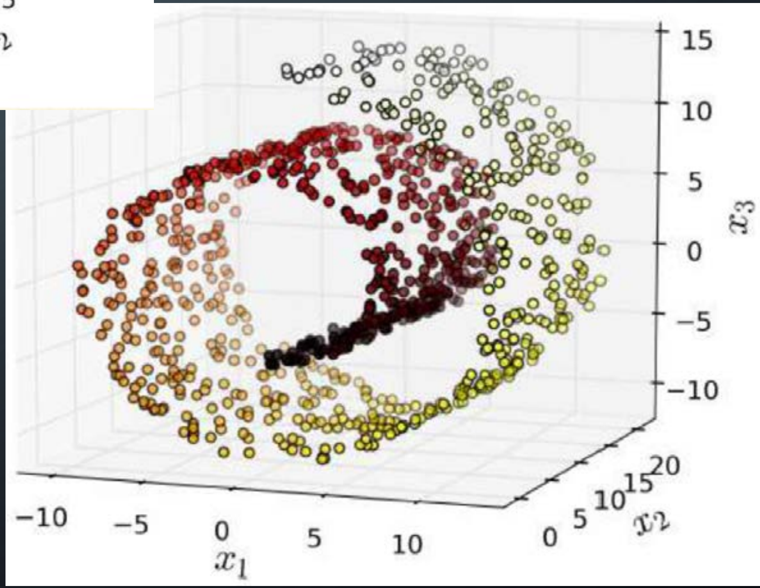
DIMENSIONALITY REDUCTION USE CASES

- Prevent overfitting
- Visualization
- Faster and more efficient models
- Storing datasets
- Decorrelation

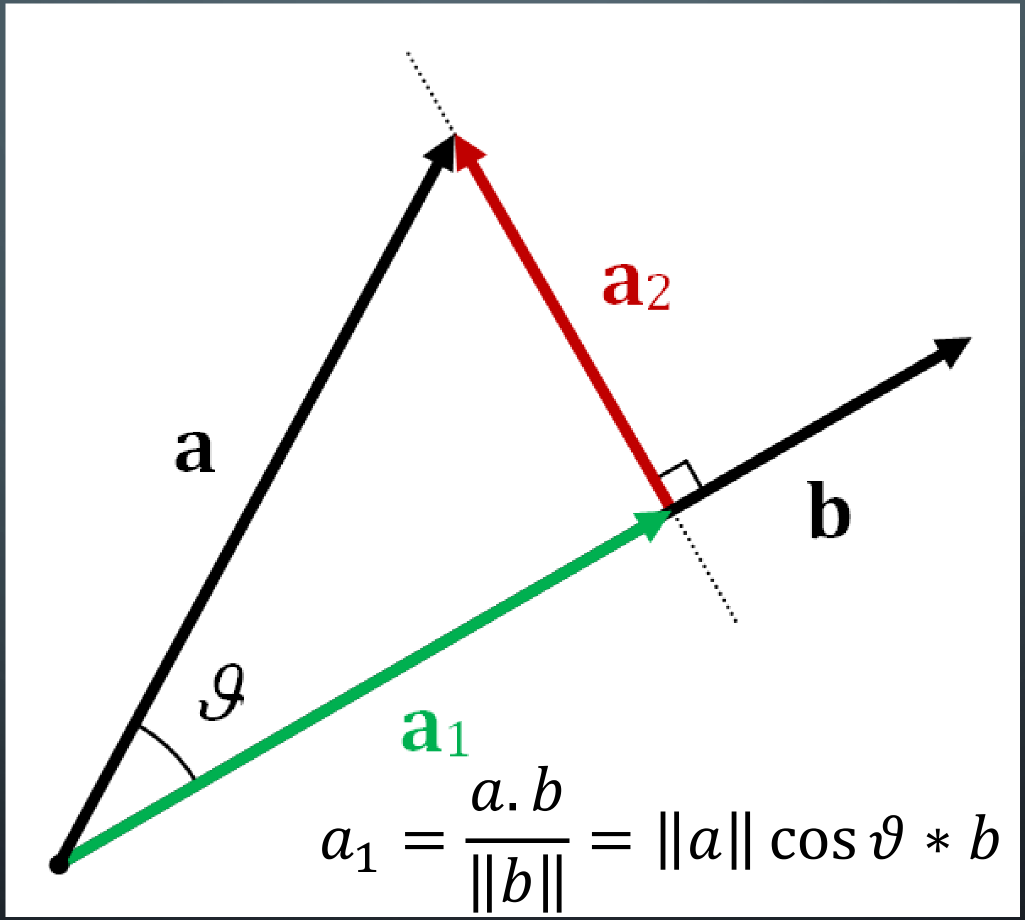
DIMENSIONALITY REDUCTION TECHNIQUES

- Feature selection
- Feature projection
  - PCA
- Manifold learning
  - LLE

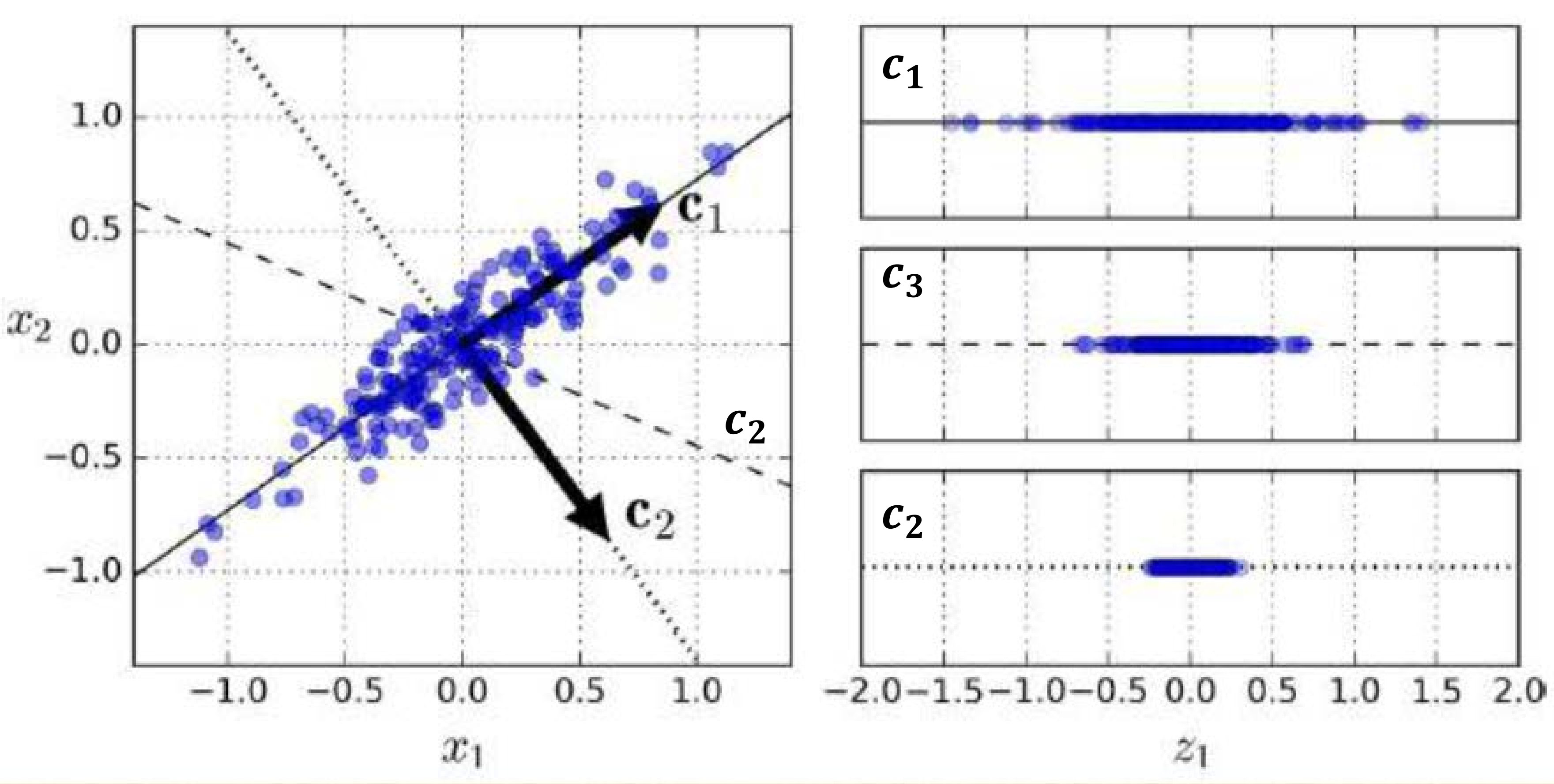




PROJECTION



PRINCIPLE COMPONENT ANALYSIS (PCA)



PCA IN VECTORS

$$M = U.\Sigma.V^T \ (SVD\ decomposition)$$

$$U^*U = UU^* = I$$

$$\Sigma\ is\ diagnal$$

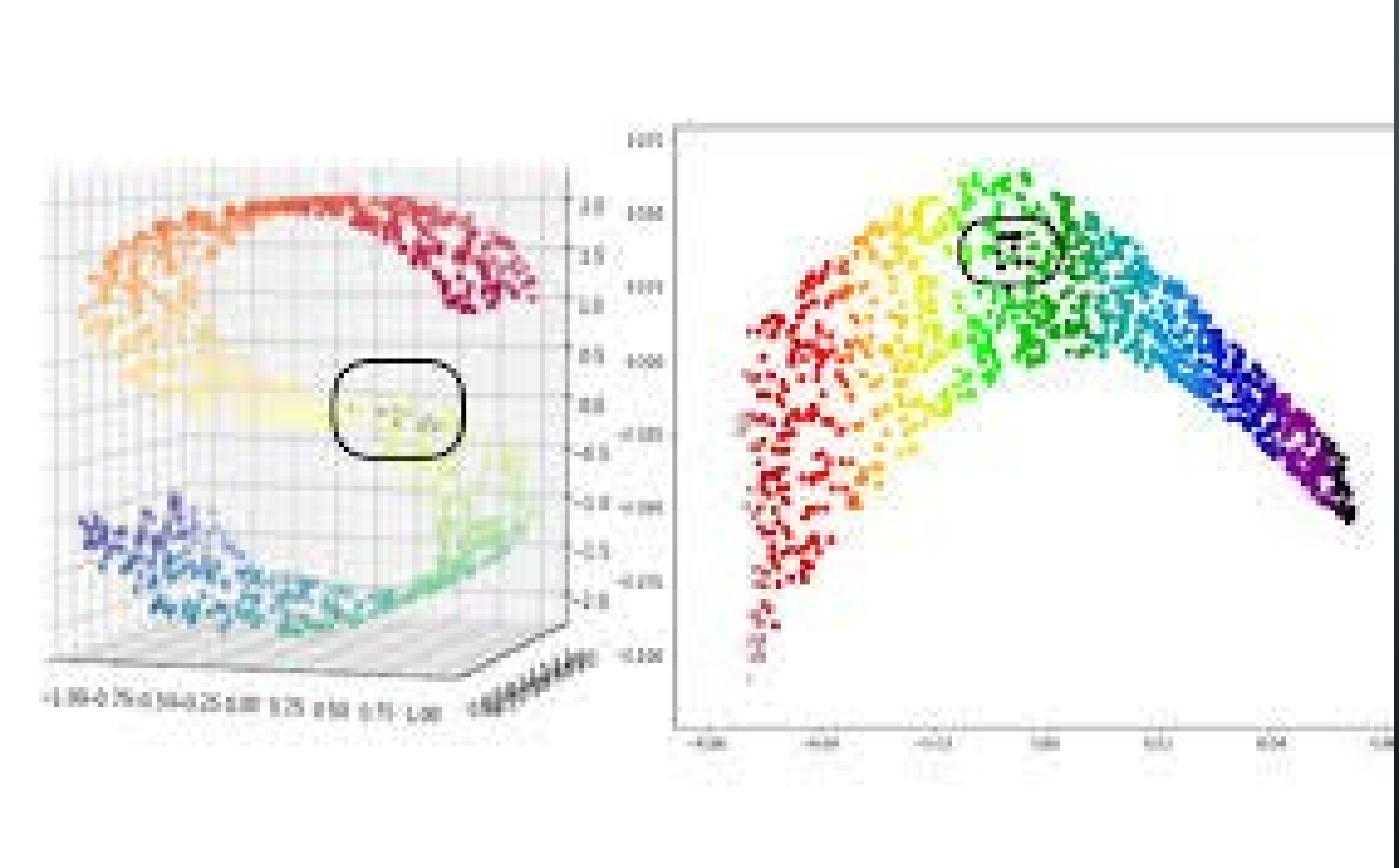
$$V is\ components$$

$$if\ M\ is\ real, U\ and\ V\ are\ real\ and\ ortogonal$$

LOCAL LINEAR EMBEDDING (LLE)

- Measuring how each training instance linearly relates to its closest neighbors
- Then looking for low-dimensional representation where these local relationships are best preserved

LLE



LLE ALGORITHM

- Step I: find k-nearest neighbors
- Step II: find W such that:

$$W = \underset{W}{\operatorname{argmin}} \sum_{i=1}^{m} \left\| X^{(i)} - \sum_{j=1}^{m} w_{i,j}\, X^{(j)} \right\|^2$$

$$W = \begin{cases} w_{i,j} & if\ X^{(j)}\ is\ not\ in\ KNN\ X^{(i)} \\ \sum_{j=1}^{m} w_{i,J} = 1 & for\ each\ i \end{cases}$$

LLE ALGORITHM

- Step III: find Z matrix in lower dimension such that:

$$Z = \underset{Z}{\operatorname{argmin}} \sum_{i=1}^{m} \left\| Z^{(i)} - \sum_{j=1}^{m} w_{i,j}\, Z^{(j)} \right\|^2$$

OTHER TECHNIQUES

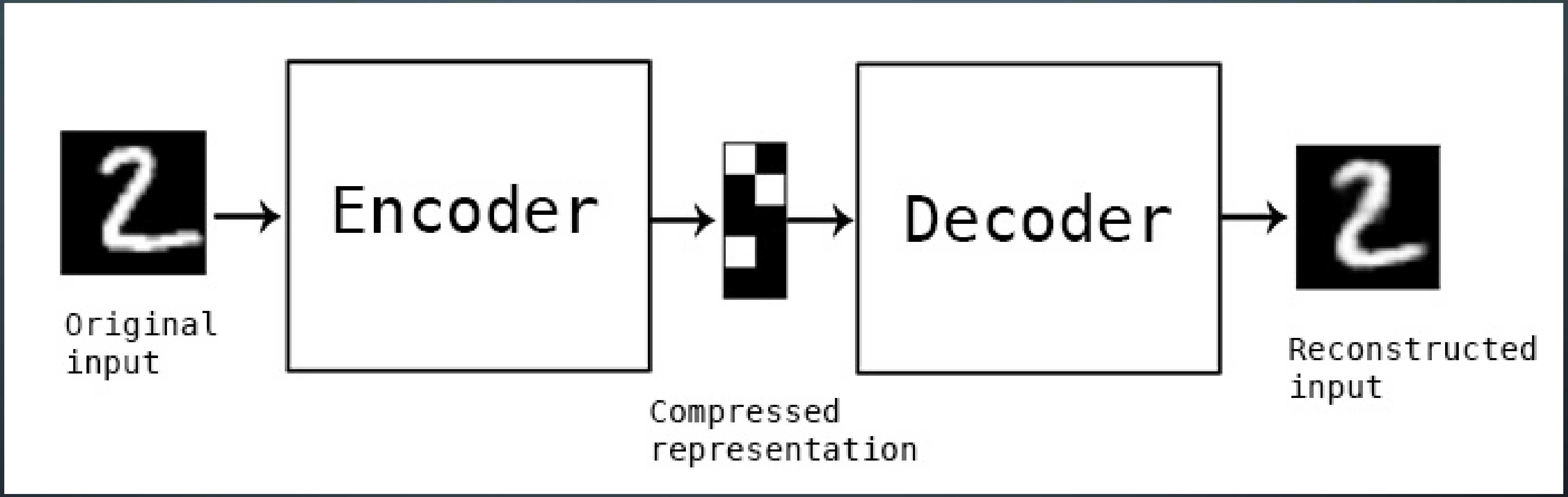
- Multidimensional scaling (MDS)
  - Preserve the distances between the instances
- Isomap
  - Forms nearest neighbor graph and try to preserve geodesic distances
- T-SNE
  - Keep similar instances close and dissimilar apart (cluster visualization)
- LDA
  - Classification algorithm, keep classes as far apart as possible.

jupyter

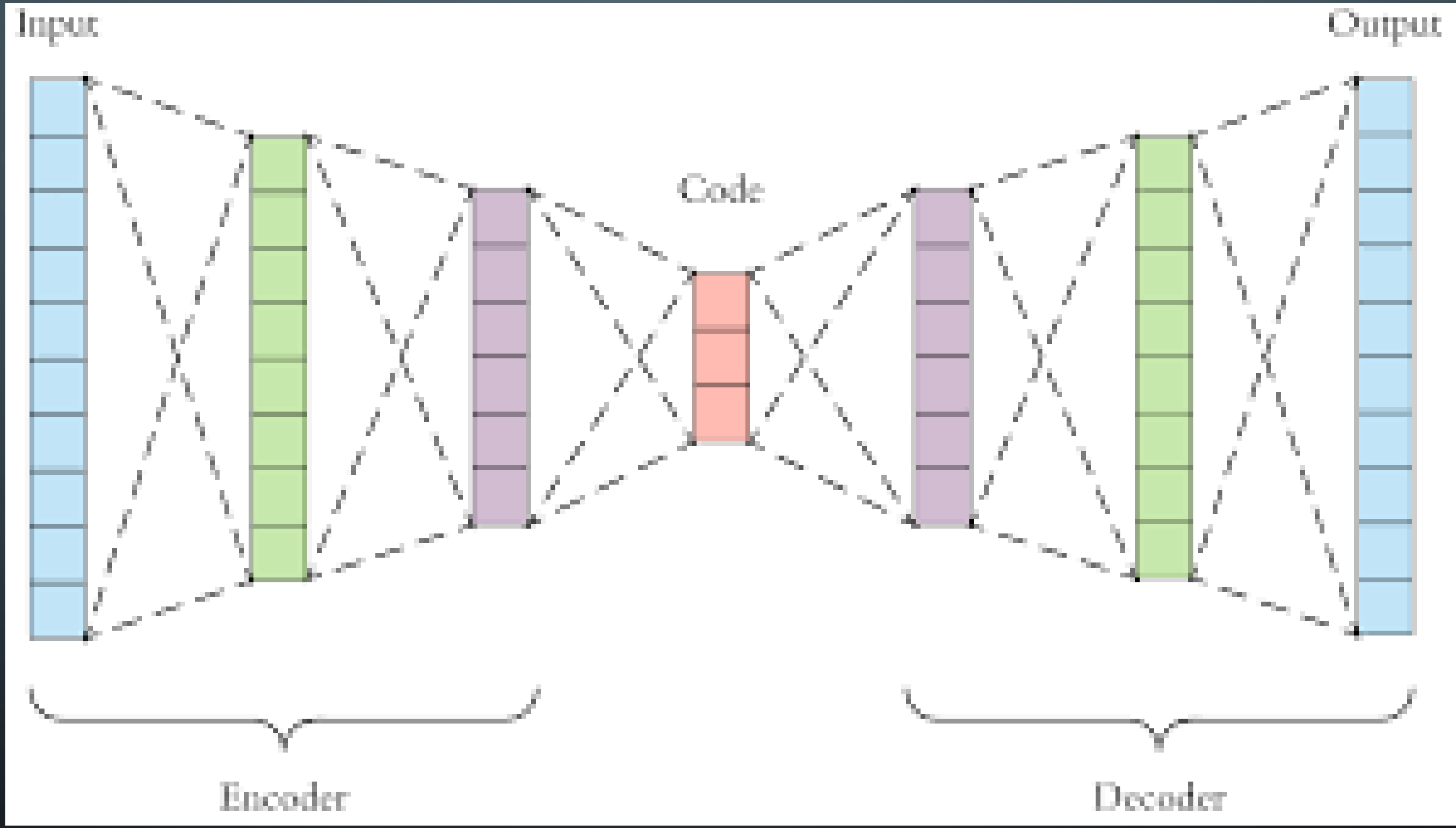
AUTOENCODERS

- Artificial Neural Network
- learns efficient data representation in an unsupervised manner
- Used in dimensional reduction

AUTOENCODERS, IDEA



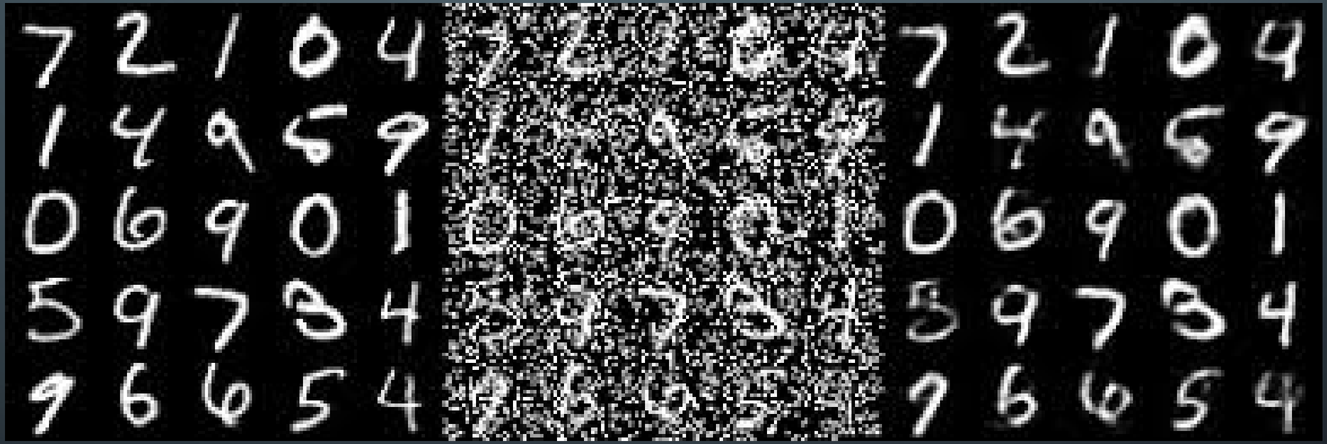
UNDERCOMPLETE AUTOENCODERS



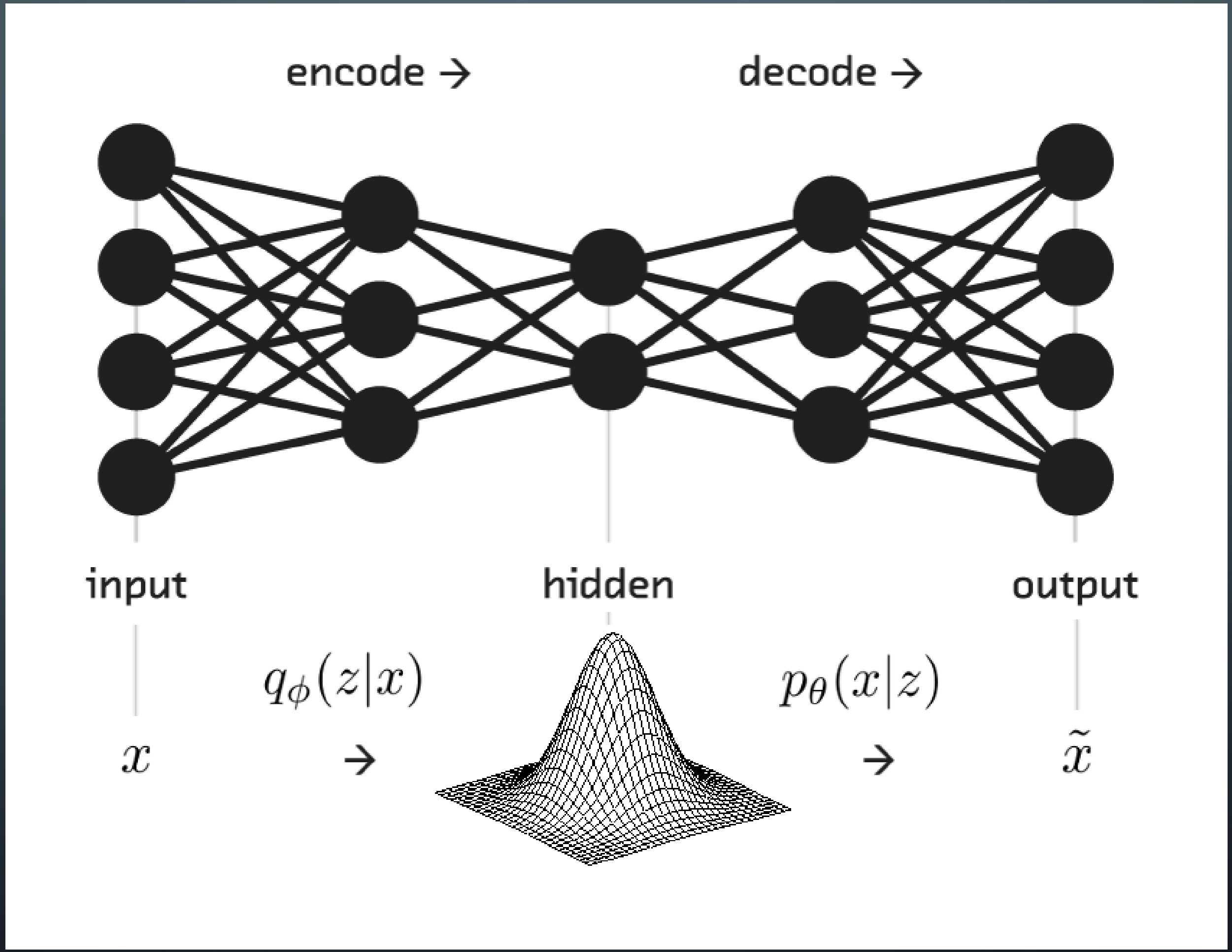
AUTOENCODERS VARIATIONS

- undercomplete Autoencoders
  - bottleneck
- Sparse Autoencoders
  - L1 regularization
  - Better for classification
- Denoising Autoencoders
  - Input is corrupted
- Variational Autoencoders
  - generates new samples

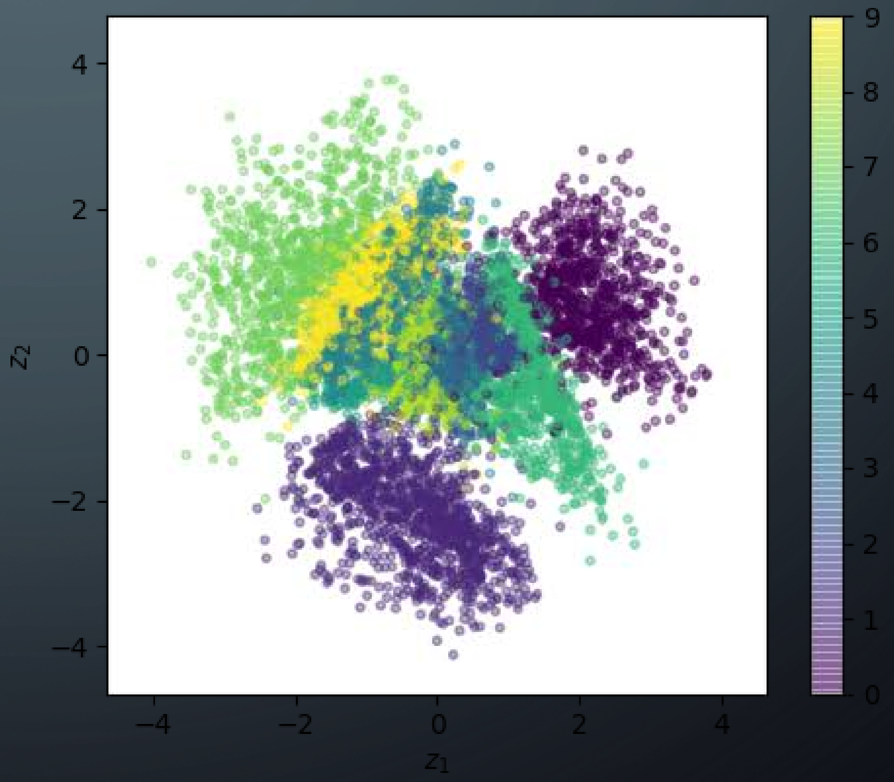
DENOISING AUTOENCODERS



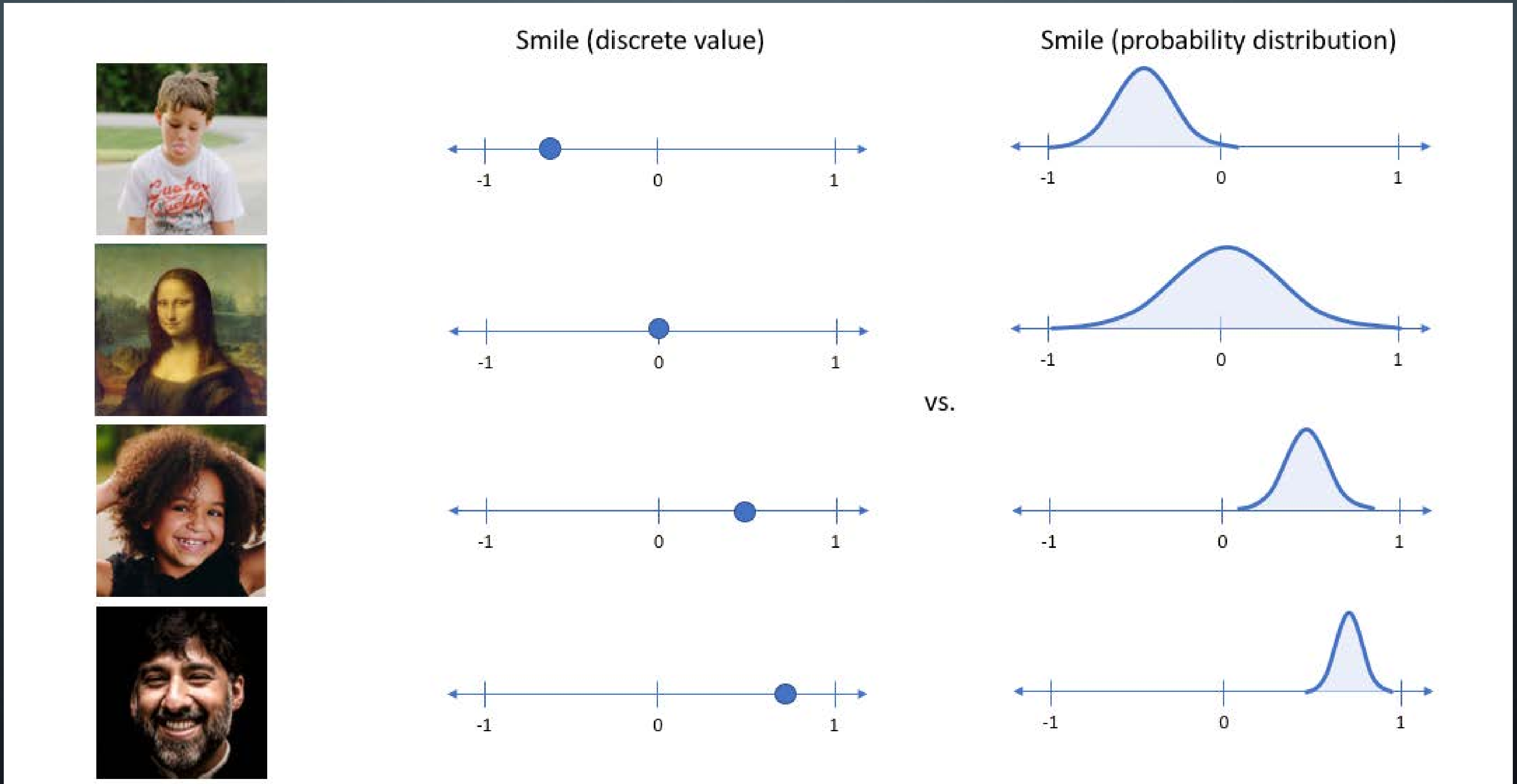
VARIATIONAL AUTOENCODERS



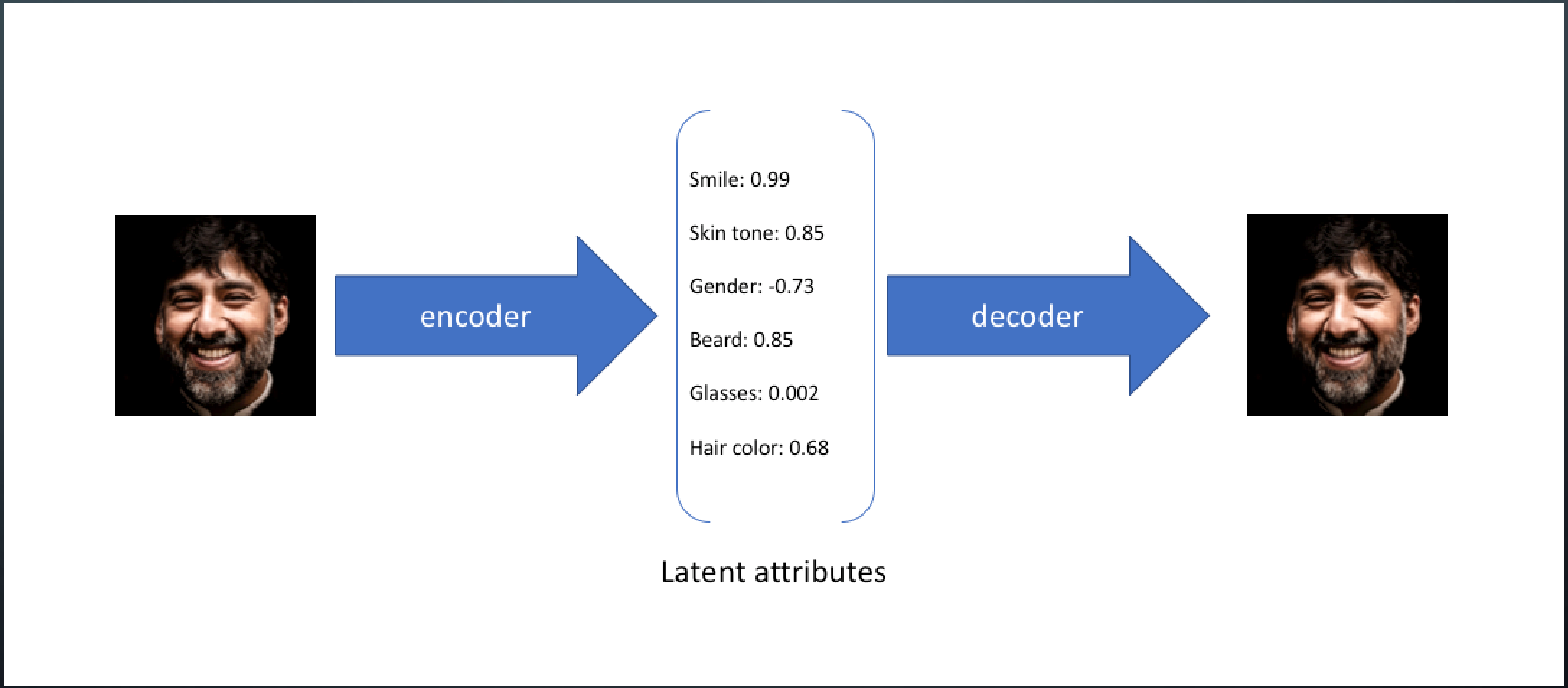
VARIATIONAL AUTOENCODERS (MNIST)



VA IDEA


Smile (discrete value)
Smile (probability distribution)
-1
0
1
vs.

NORMAL AUTOENCODERS



VARIATIONAL AUTOENCODERS