Course 3: Unsupervised Learning

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

2024-02-13

Course 3: Unsupervised Learning

Course 3: Unsupervised Learning

Summary

Last session

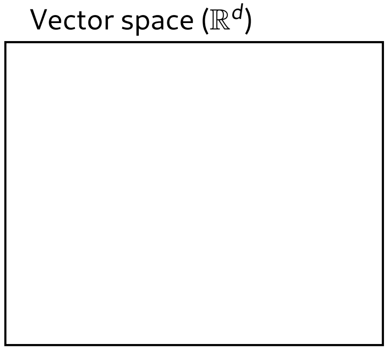
1. Supervised learning - learning from labeled examples
2. Bias/variance tradeoff
3. Overfitting and cross-validation
4. VC Dimension and curse of dimensionality

Today's session

1. Learning from Unlabeled examples
2. Clustering, decomposition and dimensionality reduction

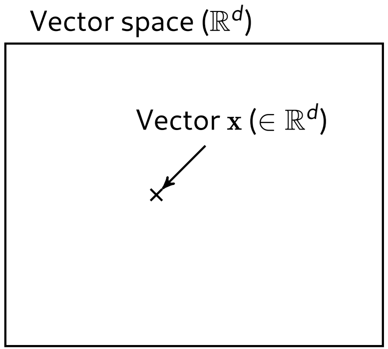
Summary
Last session
1. Supervised learning - learning from labeled examples
2. Bias/variance tradeoff
3. Overfitting and cross-validation
4. VC Dimension and curse of dimensionality
Today's session
1. Learning from Unlabeled examples
2. Clustering, decomposition and dimensionality reduction

Notations

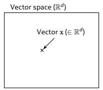


Notations
Vector space ($\mathbb{R}^d$)

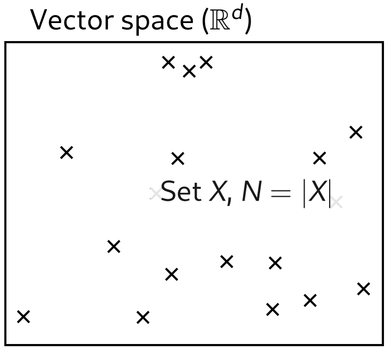
Notations



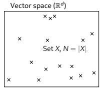
Notations



Notations



Notations



Goal

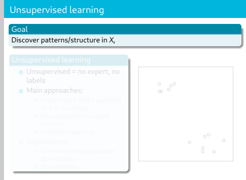
Discover patterns/structure in X,

Unsupervised learning

- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization...

2024-02-13 Course 3: Unsupervised Learning

└─Unsupervised learning



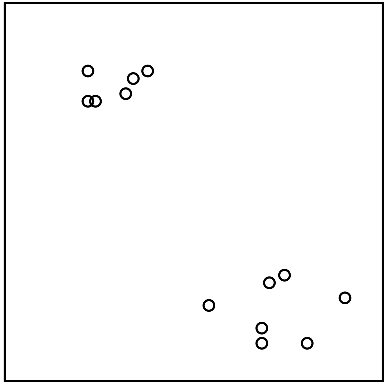
Unsupervised learning

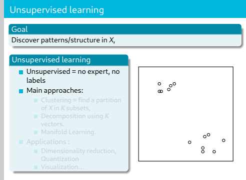
Goal

Discover patterns/structure in X,

Unsupervised learning

- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization...



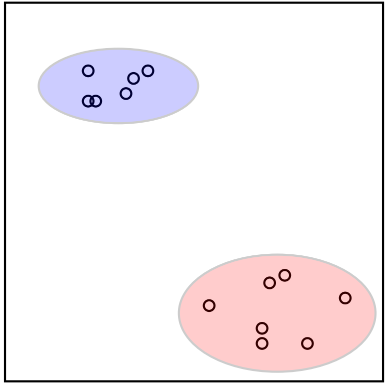


Goal

Discover patterns/structure in X,

Unsupervised learning

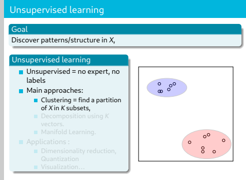
- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization…



2024-02-13

Course 3: Unsupervised Learning

└Unsupervised learning



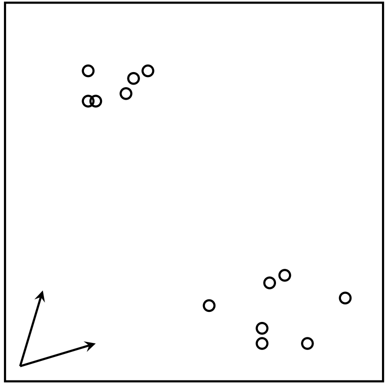
Unsupervised learning

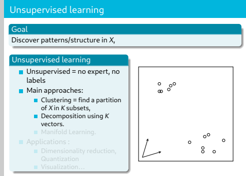
Goal

Discover patterns/structure in X,

Unsupervised learning

- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization...





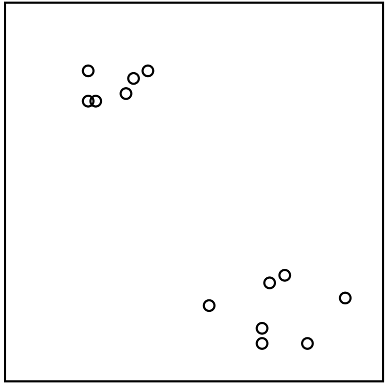
Unsupervised learning

Goal

Discover patterns/structure in X,

Unsupervised learning

- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization...



Unsupervised learning
Goal
Discover patterns/structure in X,
Unsupervised learning
Unsupervised = no expert, no labels
Main approaches:
Clustering = find a partition of X in K subsets,
Decomposition using K vectors.
Manifold Learning.
Applications :
Dimensionality reduction, Quantization
Visualization...

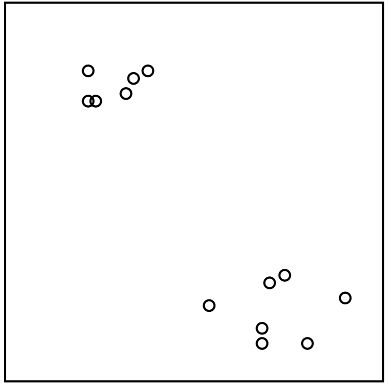
Unsupervised learning

Goal

Discover patterns/structure in X,

Unsupervised learning

- Unsupervised = no expert, no labels
- Main approaches:
  - Clustering = find a partition of X in K subsets,
  - Decomposition using K vectors.
  - Manifold Learning.
- Applications :
  - Dimensionality reduction, Quantization
  - Visualization...

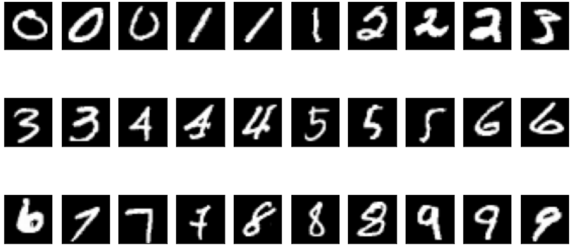


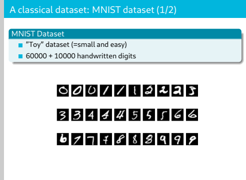
Unsupervised learning
Goal
Discover patterns/structure in X,
Unsupervised learning
Unsupervised = no expert, no labels
Main approaches:
Clustering = find a partition of X in K subsets,
Decomposition using K vectors.
Manifold Learning.
Applications :
Dimensionality reduction, Quantization
Visualization...

A classical dataset: MNIST dataset (1/2)

MNIST Dataset

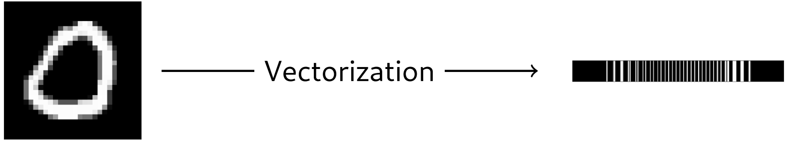
- "Toy" dataset (=small and easy)
- 60000 + 10000 handwritten digits





Let's look at an example that looks a little bit more like real data. The MNIST dataset is small dataset of handwritten digits. It used to be an important benchmark, but it is considered too easy today to be a serious machine learning benchmark, so that is why we say it is a "toy" dataset. MNIST is composed of 60000 examples of digits that are used for training, and 10000 that are used for test.

A classical dataset: MNIST dataset (2/2)

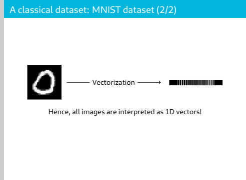


Hence, all images are interpreted as 1D vectors!

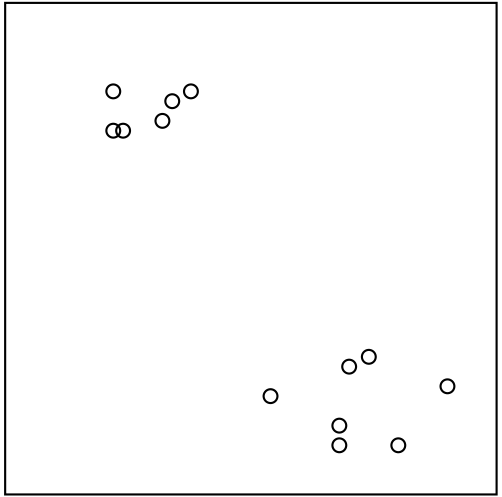
2024-02-13

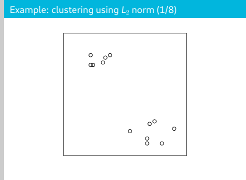
Course 3: Unsupervised Learning

└A classical dataset: MNIST dataset (2/2)

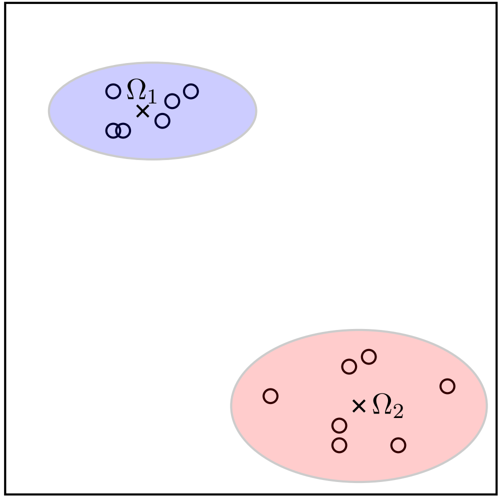


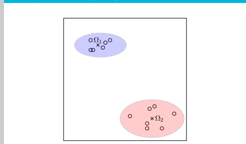
Example: clustering using L_2 norm (1/8)





Here is a visual example. If we have the following set of points, then the following two centroids Ω_1 and Ω_2 would be reasonable candidates for a clustering with two clusters.





Here is a visual example. If we have the following set of points, then the following two centroids Ω_1 and Ω_2 would be reasonable candidates for a clustering with two clusters.

Example: clustering using L_2 norm (2/8)

An example to perform clustering is to rely on distances to centroids.
We define K *cluster centroids* $\Omega_k, \forall k \in [1..K]$.
Here, each vector is associated with the cluster whose centroid is of minimal distance.

Definitions

We denote $q : \mathbb{R}^d \rightarrow [1..K]$ a function that associates a vector $\mathbf{x}$ with the index of (one of) its closest centroid $q(\mathbf{x})$. Formally:

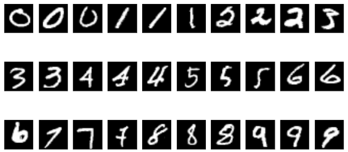
- $\forall k \in [1..K], \Omega_k \in \mathbb{R}^d$
- $\forall \mathbf{x} \in X, \forall j \in [1..K], \|\mathbf{x} - \Omega_{q(\mathbf{x})}\|_2 \leq \|\mathbf{x} - \Omega_j\|_2$
- Error $E(q) \triangleq \sum_{\mathbf{x} \in X} \|\mathbf{x} - \Omega_{q(\mathbf{x})}\|_2$
- $X = \bigcup_k \underbrace{\{\mathbf{x} \in X, q(\mathbf{x}) = k\}}_{\text{cluster } k}$

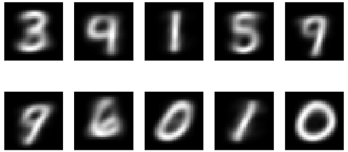
Example: clustering using L_2 norm (2/8)
An example to perform clustering is to rely on distances to centroids. We define K *cluster centroids* $\Omega_k, \forall k \in [1..K]$. Here, each vector is associated with the cluster whose centroid is of minimal distance.
Definitions
We denote $q : \mathbb{R}^d \rightarrow [1..K]$ a function that associates a vector $\mathbf{x}$ with the index of (one of) its closest centroid $q(\mathbf{x})$. Formally:
$\forall k \in [1..K], \Omega_k \in \mathbb{R}^d$
$\forall \mathbf{x} \in X, \forall j \in [1..K], \|\mathbf{x} - \Omega_{q(\mathbf{x})}\|_2 \leq \|\mathbf{x} - \Omega_j\|_2$
Error $E(q) \triangleq \sum_{\mathbf{x} \in X} \|\mathbf{x} - \Omega_{q(\mathbf{x})}\|_2$
$X = \bigcup_k \underbrace{\{\mathbf{x} \in X, q(\mathbf{x}) = k\}}_{\text{cluster } k}$

Here, we provide a formal definition of clustering using centroids. Note that there are other ways to define clustering, using regions, using density of spaces, using probabilities, etc...
The second point is the way to define the closest centroid.
The important point to note here is the definition of the error, which can be defined as the sum of all distances between points and their closest cluster centroid.

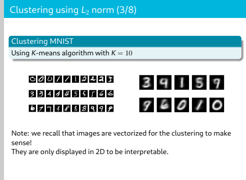
Clustering MNIST

Using K-means algorithm with $K = 10$





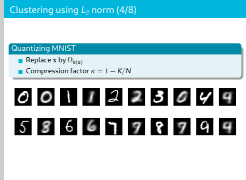
Note: we recall that images are vectorized for the clustering to make sense!
They are only displayed in 2D to be interpretable.



Let's look at an example that looks a little bit more like real data. The MNIST dataset is small dataset of handwritten digits. It used to be an important benchmark, but it is considered too easy today to be a serious machine learning benchmark, so that is why we say it is a "toy" dataset.
MNIST is composed of 60000 examples of digits that are used for training, and 10000 that are used for test.
We can do a simple clustering test on this dataset, by using the K-Means algorithm.
Briefly, the K-means algorithm iterates between (a) assigning each point to a cluster by considering the distance to centroids, and (b) calculating the centroids for the next iteration by computing the average in each cluster. Centroid clusters can be initiliazed randomly.
The K-means algorithm stops when a certain criterion is met (number of iterations, or difference between iterations is small enough).
See here `https://upload.wikimedia.org/wikipedia/commons/f/fb/K-means.png` (picture is nice) or `https://en.wikipedia.org/wiki/K-means_clustering`
Maybe a very quick explanation of Kmeans on the board is good if the time enables it.
The bottom left figure represent original examples of MNIST. The bottom right figure shows the obtained cluster centroids with Kmeans. We can comment that some of the clusters seem to capture one digit (6, 1, 2, 0), but that other digits can correspond to several clusters (8, 4, 3).
The next figure will illustrate this more precisely.

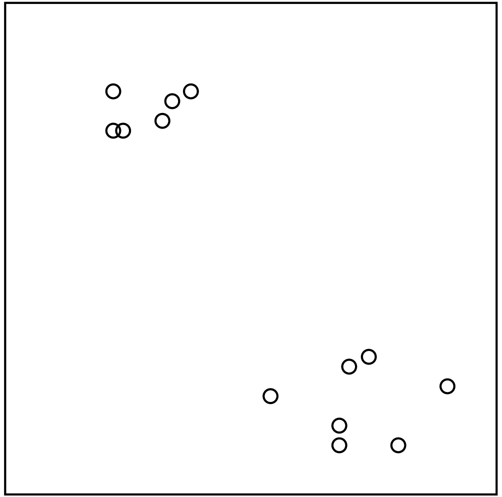
Quantizing MNIST

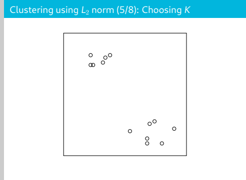
- Replace $\mathbf{x}$ by $\Omega_{k(\mathbf{x})}$
- Compression factor $\kappa = 1 - K/N$



We have chosen here a random example of each digit, and we show the closest cluster centroid. We see that there are issues with 3, 4, 5, 7 and 8, even though we have tried to find 10 clusters.
In the top part of the slide, we also explain that we can actually use Clustering for compression; we just have to store the centroids, and the cluster label.

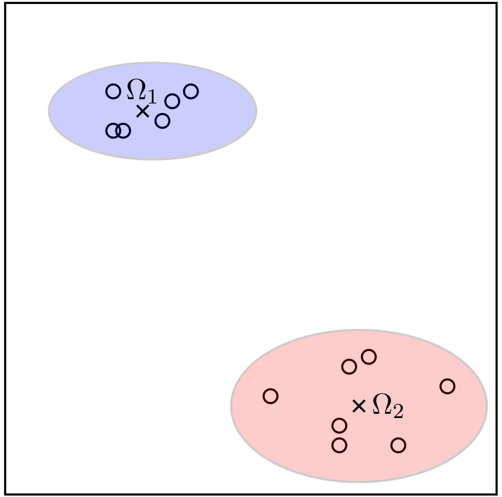
Clustering using L_2 norm (5/8): Choosing K



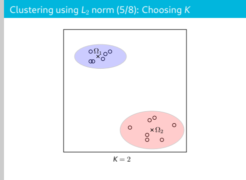


Changing the number of centroids changes the clustering... And the signification of clusters.

Clustering using L_2 norm (5/8): Choosing K

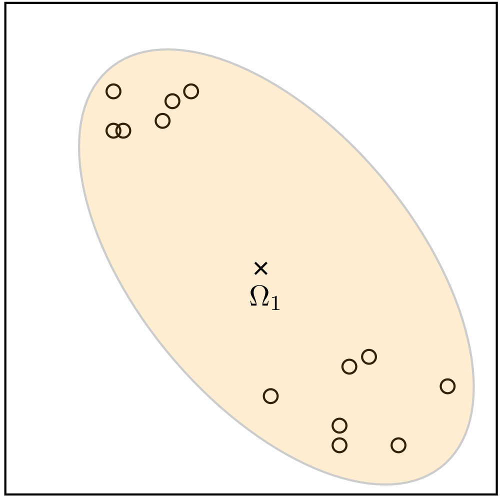


$K = 2$

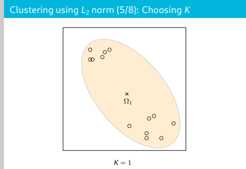


Changing the number of centroids changes the clustering... And the signification of clusters.

Clustering using L_2 norm (5/8): Choosing K

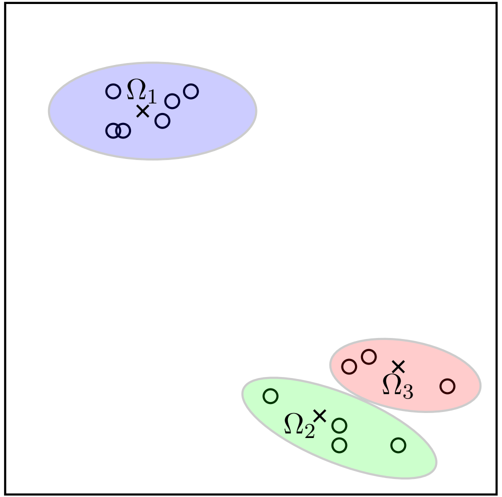


$K = 1$

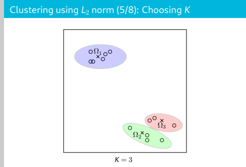


Changing the number of centroids changes the clustering... And the signification of clusters.

Clustering using L_2 norm (5/8): Choosing K

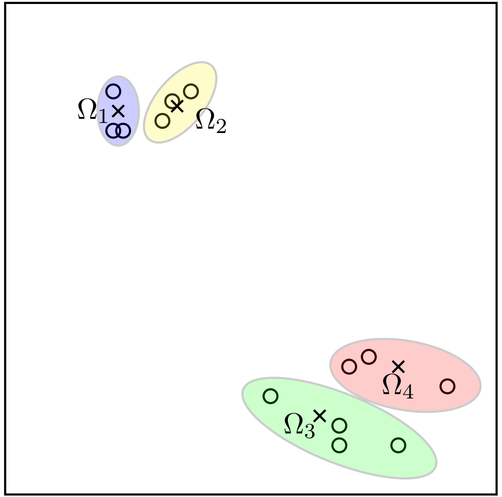


$K = 3$

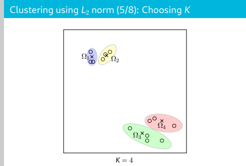


Changing the number of centroids changes the clustering... And the signification of clusters.

Clustering using L_2 norm (5/8): Choosing K

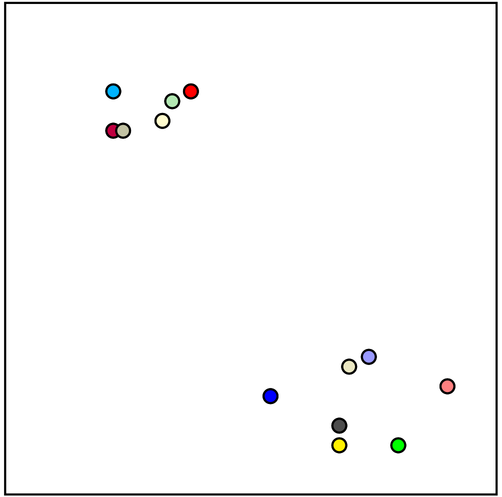


$K = 4$

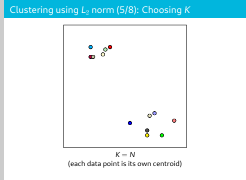


Changing the number of centroids changes the clustering... And the signification of clusters.

Clustering using L_2 norm (5/8): Choosing K



$K = N$
(each data point is its own centroid)

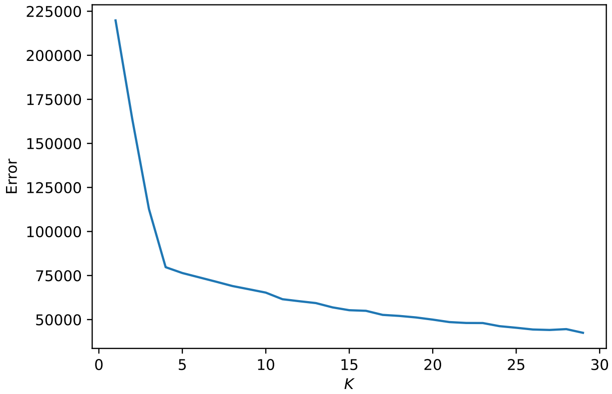


Changing the number of centroids changes the clustering... And the signification of clusters.

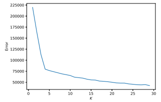
Clustering using L_2 norm (6/8)

Choosing K

- Finding a compromise between error and compression,
- Simple practical method : "elbow".



Clustering using L_2 norm (6/8)
Choosing K
- Finding a compromise between error and compression,
- Simple practical method : "elbow".



It is important to say that this is the ideal case! Here, we clearly see a value of K after which it is not necessary to add more clusters.

Clustering using L_2 norm (7/8)

Optimal clustering

- Define $E_{opt_K}(q^*) \triangleq \arg \min_{q:\mathbb{R}^d \to [1..K]} E(q)$,
- Finding an optimal clustering is an NP-hard problem.

Properties

- $0 = E_{opt_N}(q^*) \leq E_{opt_{N-1}}(q^*) \leq \cdots \leq E_{opt_1}(q^*) = var(X)$,
  - Proof: monotonicity by particularization, extremes with identity function (left) and variance (right).
- $0 \leq K \leq \frac{N-1}{N}$.

Changing the number of centroids changes the clustering... And the signification of clusters.

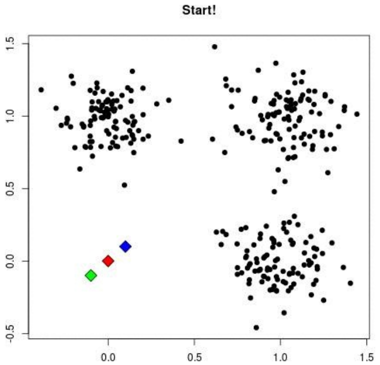
Clustering using L_2 norm (7/8)
Optimal clustering
Define $E_{opt_K}(q^*) \triangleq \arg \min_{q:\mathbb{R}^d \to [1..K]} E(q)$,
Finding an optimal clustering is an NP-hard problem.
Properties
$0 = E_{opt_N}(q^*) \leq E_{opt_{N-1}}(q^*) \leq \cdots \leq E_{opt_1}(q^*) = var(X)$,
Proof: monotonicity by particularization, extremes with identity function (left) and variance (right).
$0 \leq K \leq \frac{N-1}{N}$.
Changing the number of centroids changes the clustering... And the signification of clusters.

About the properties :
On the left side, if we take a cluster for each point in the space (N cluster centroids), then obviously the error is 0.
On the right side, if we take only one cluster, then the best cluster that can be chosen is the average of all points, in which case the error is exactly the variance across X.

K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



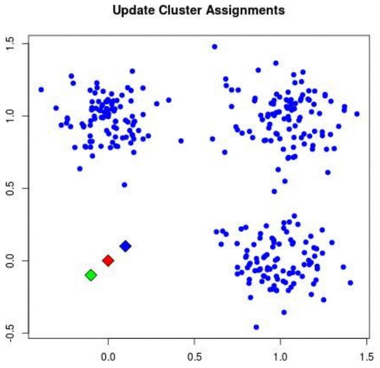
Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13 Course 3: Unsupervised Learning
Clustering using L_2 norm (8/8)

K-means algorithm

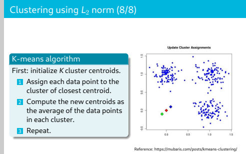
First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13 Course 3: Unsupervised Learning
└─Clustering using L_2 norm (8/8)

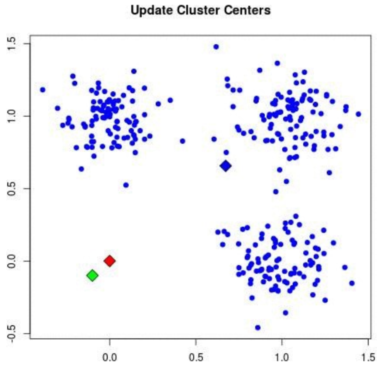


Clustering using L_2 norm (8/8)

K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.

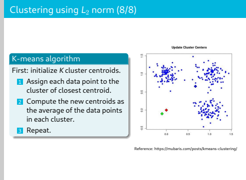


Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13

Course 3: Unsupervised Learning

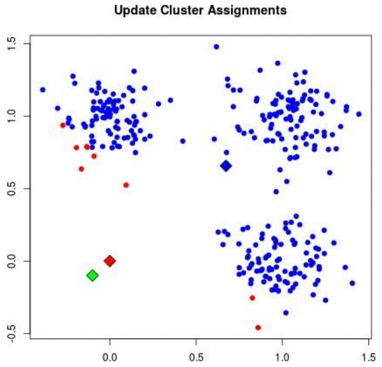
└Clustering using L_2 norm (8/8)



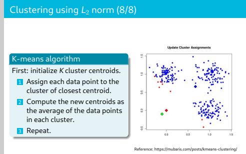
K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



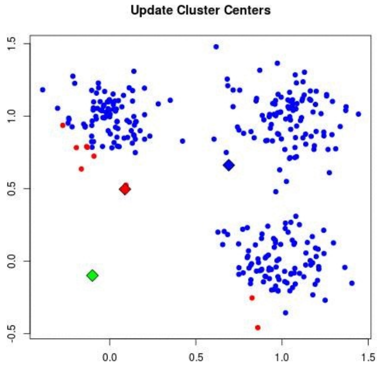
Reference: https://mubaris.com/posts/kmeans-clustering/



K-means algorithm

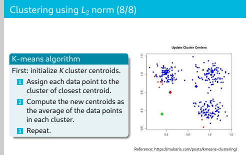
First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



Reference: https://mubaris.com/posts/kmeans-clustering/

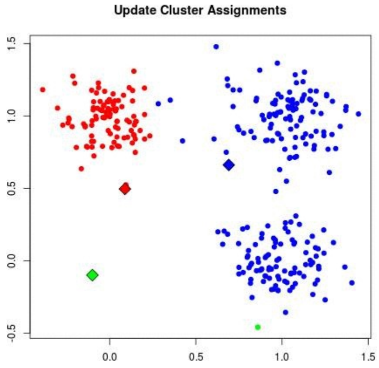
2024-02-13 Course 3: Unsupervised Learning
└Clustering using L_2 norm (8/8)



K-means algorithm

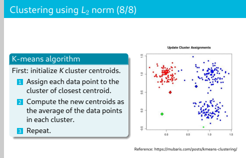
First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



Reference: https://mubaris.com/posts/kmeans-clustering/

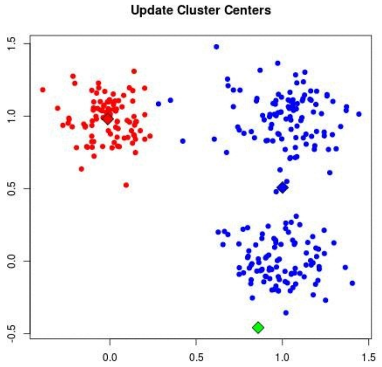
2024-02-13
Course 3: Unsupervised Learning
└Clustering using L_2 norm (8/8)



K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.

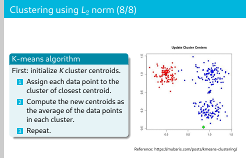


Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13

Course 3: Unsupervised Learning

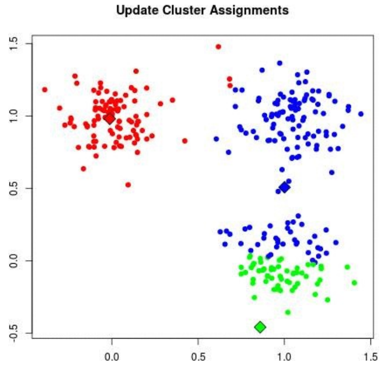
└─Clustering using L_2 norm (8/8)



K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.

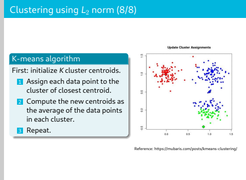


Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13

Course 3: Unsupervised Learning

└─Clustering using L_2 norm (8/8)

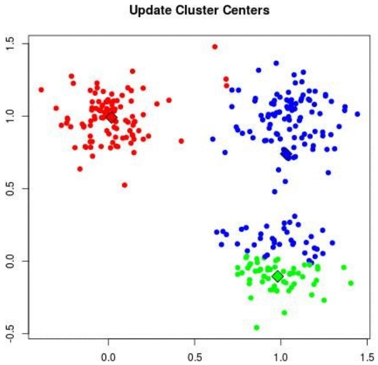


Clustering using L_2 norm (8/8)

K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.

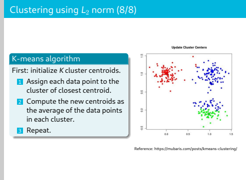


Reference: https://mubaris.com/posts/kmeans-clustering/

2024-02-13

Course 3: Unsupervised Learning

└─Clustering using L_2 norm (8/8)

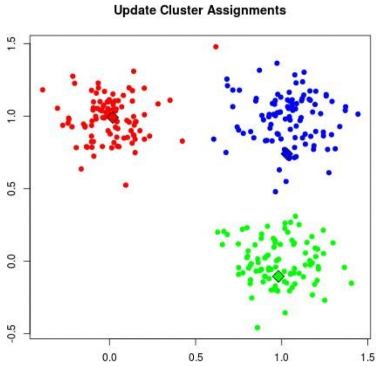


Clustering using L_2 norm (8/8)

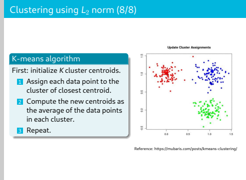
K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



Reference: https://mubaris.com/posts/kmeans-clustering/

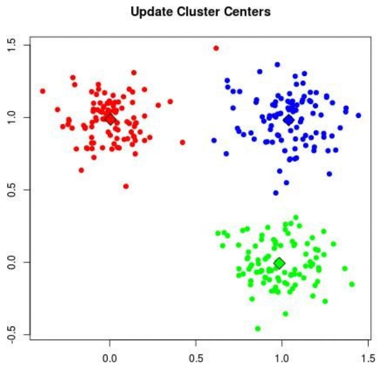


Clustering using L_2 norm (8/8)

K-means algorithm

First: initialize K cluster centroids.

1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



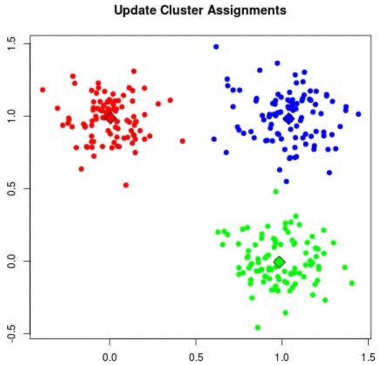
Reference: https://mubaris.com/posts/kmeans-clustering/

└─Clustering using L_2 norm (8/8)

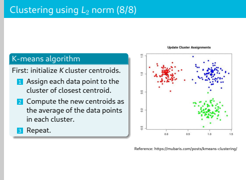
K-means algorithm

First: initialize K cluster centroids.

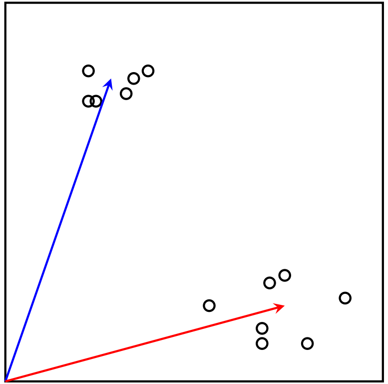
1. Assign each data point to the cluster of closest centroid.
2. Compute the new centroids as the average of the data points in each cluster.
3. Repeat.



Reference: https://mubaris.com/posts/kmeans-clustering/

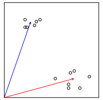


Example 2: Sparse Dictionary Learning (1/5)

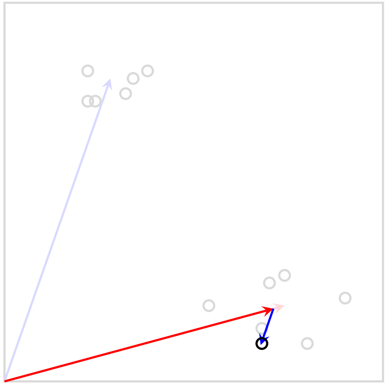


Example 2: Sparse Dictionary Learning (1/5)

Example 2: Sparse Dictionary Learning (1/5)

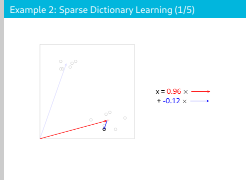


Example 2: Sparse Dictionary Learning (1/5)

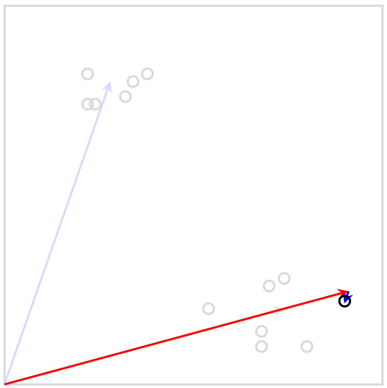


$x = 0.96 \times \longrightarrow$

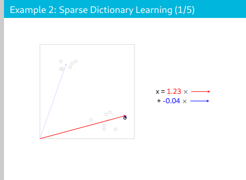
$+\ -0.12 \times \longrightarrow$



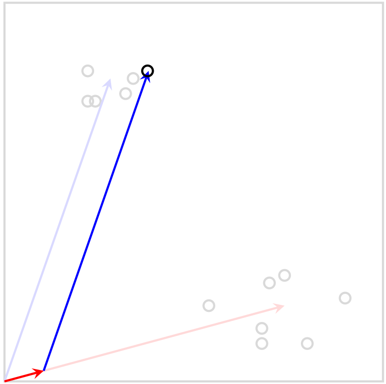
Example 2: Sparse Dictionary Learning (1/5)



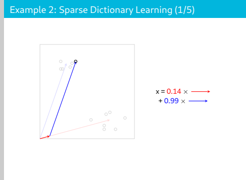
$x = {\color{red}1.23} \times {\color{red}\longrightarrow}$
$+ {\color{blue}-0.04} \times {\color{blue}\longrightarrow}$



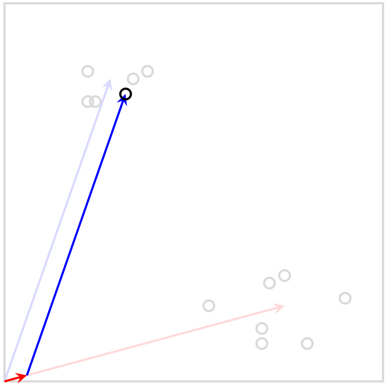
Example 2: Sparse Dictionary Learning (1/5)



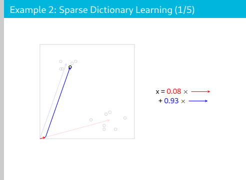
$$x = 0.14 \times \longrightarrow$$
$$+\ 0.99 \times \longrightarrow$$



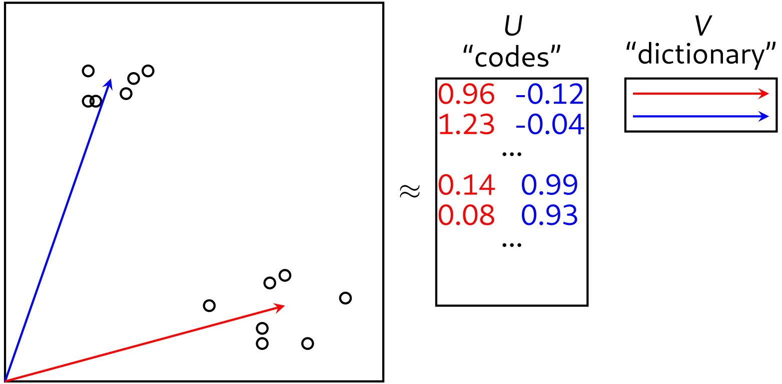
Example 2: Sparse Dictionary Learning (1/5)



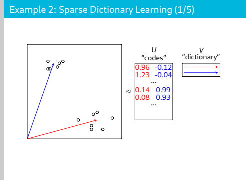
$$x = \color{red}{0.08} \times \color{red}{\longrightarrow}$$
$$+\ \color{blue}{0.93} \times \color{blue}{\longrightarrow}$$



Example 2: Sparse Dictionary Learning (1/5)



Example 2: Sparse Dictionary Learning (1/5)

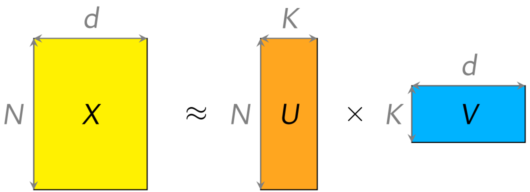


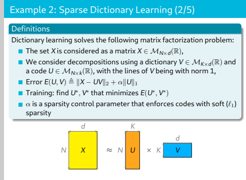
Example 2: Sparse Dictionary Learning (2/5)

Definitions

Dictionary learning solves the following matrix factorization problem:

- The set X is considered as a matrix $X \in \mathcal{M}_{N\times d}(\mathbb{R})$,
- We consider decompositions using a dictionary $V \in \mathcal{M}_{K\times d}(\mathbb{R})$ and a code $U \in \mathcal{M}_{N\times k}(\mathbb{R})$, with the lines of V being with norm 1,
- Error $E(U, V) \triangleq \|X - UV\|_2 + \alpha\|U\|_1$
- Training: find U^*, V^* that minimizes $E(U^*, V^*)$
- α is a sparsity control parameter that enforces codes with soft (ℓ_1) sparsity



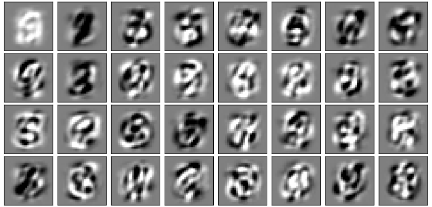


Here, just unroll the definition, by saying that Dictionary Learning is one way (among others) to perfom matrix factorization. It takes advantage of targetting a sparse code U. We will not explain here how to solve the optimization problem.

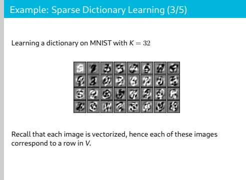
Note that the definition of the error here includes the sparsity term. As a consequence, formally the error defined here is the optimization problem that is being solved, while the error (of reconstruction) regarding the original data is only the first term with the L2 norm.

Example: Sparse Dictionary Learning (3/5)

Learning a dictionary on MNIST with $K = 32$



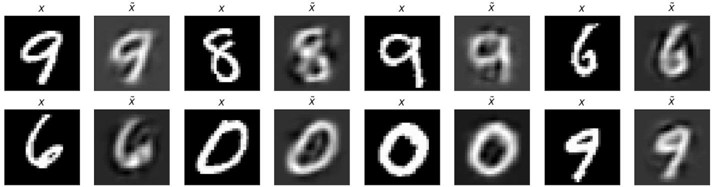
Recall that each image is vectorized, hence each of these images correspond to a row in V.

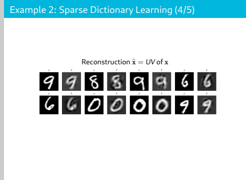


This is what a sparse dictionnary looks like, with 64 atoms in the dictionnary, on MNIST.

Example 2: Sparse Dictionary Learning (4/5)

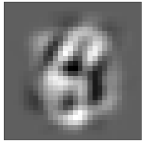
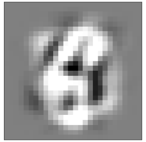
Reconstruction $\tilde{\mathbf{x}} = UV$ of $\mathbf{x}$

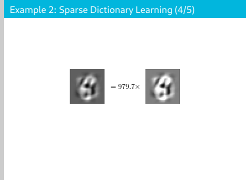




In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

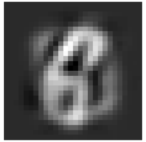
Example 2: Sparse Dictionary Learning (4/5)

 $= 979.7 \times$ 

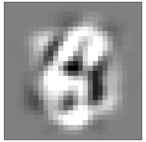


In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

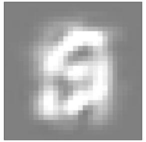
Example 2: Sparse Dictionary Learning (4/5)

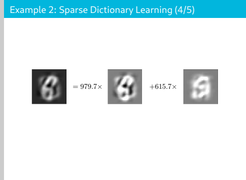


$= 979.7\times$



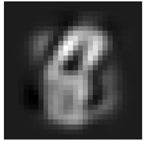
$+615.7\times$



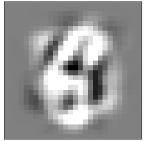


In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

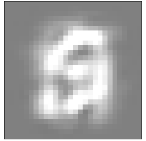
Example 2: Sparse Dictionary Learning (4/5)



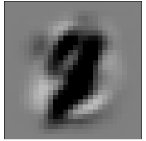
$= 979.7\times$

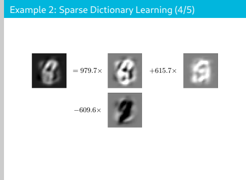


$+615.7\times$



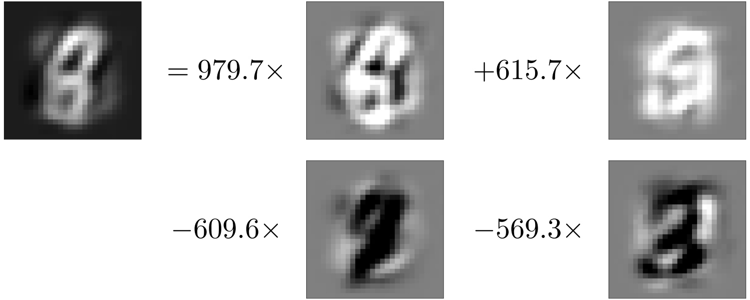
$-609.6\times$

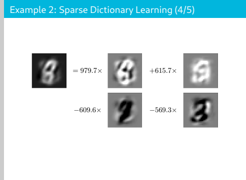




In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

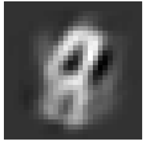
Example 2: Sparse Dictionary Learning (4/5)



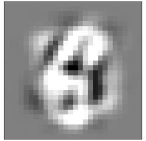


In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

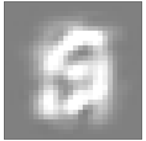
Example 2: Sparse Dictionary Learning (4/5)



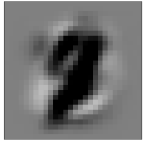
$= 979.7\times$



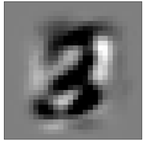
$+615.7\times$



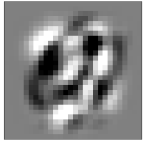
$-609.6\times$

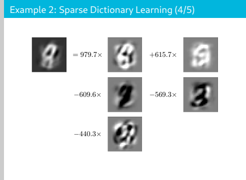


$-569.3\times$



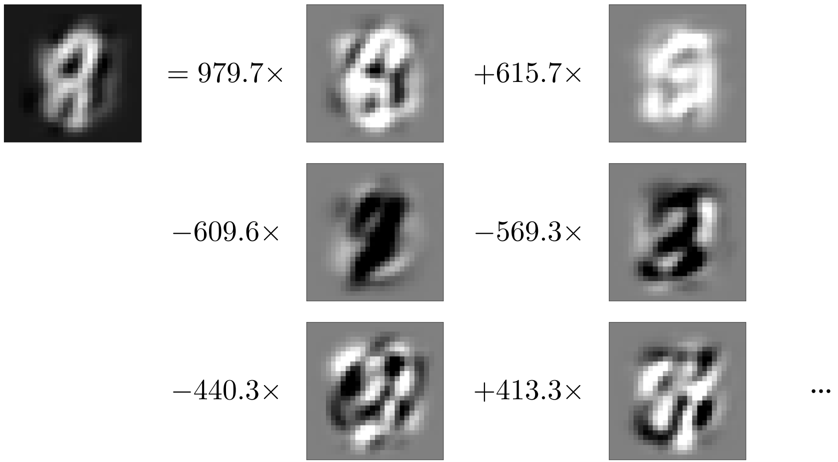
$-440.3\times$

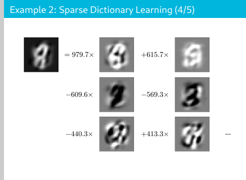




In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

Example 2: Sparse Dictionary Learning (4/5)

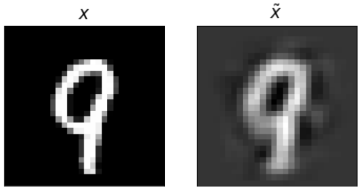


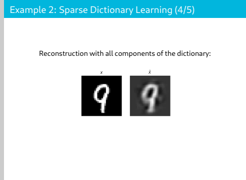


In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

Example 2: Sparse Dictionary Learning (4/5)

Reconstruction with all components of the dictionary:





In this slide we show the result of reconstructing the original vectors using the learnt dictionnary. In the first slide, we only show the results of reconstruction. In the following slides, we unroll the decomposition of a particular example, by adding the combination of the most important atoms (highest absolute values) (i.e. how the atoms are weighted to reconstruct the original vector).

Example 2: Sparse Dictionary Learning (5/5)

Optimal error

- $E_{opt_K}(U^*, V^*) \triangleq \arg\min_{U,V} E(U, V)$.

Some results

- For $\alpha = 0$ and $K \geq d$, $E_{opt_d}(U^*, V^*) = 0$,
  - One can choose any completion of a basis.
- For $K = N$, $\forall\alpha$, $E_{opt_K}(U^*, V^*) = \alpha N$,
  - If vectors of X are with norm 1, one can choose $V = X$ and $U = \mathbf{I}_N$.

Example 2: Sparse Dictionary Learning (5/5)

Optimal error

- $E_{opt_K}(U^*, V^*) \triangleq \arg\min_{U,V} E(U, V)$.

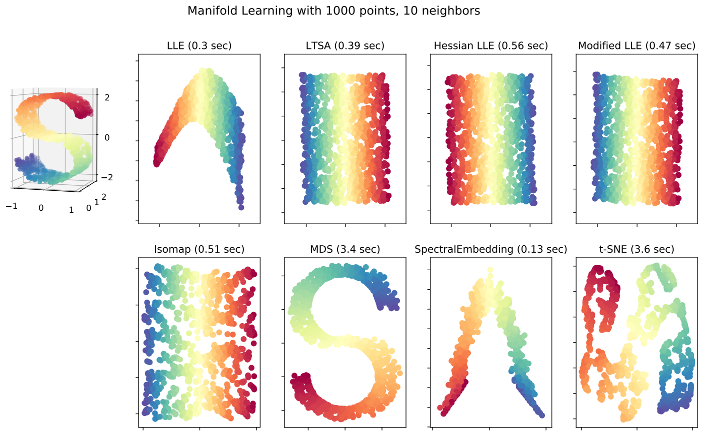
Some results

- For $\alpha = 0$ and $K \geq d$, $E_{opt_d}(U^*, V^*) = 0$,
  - One can choose any completion of a basis.
- For $K = N$, $\forall\alpha$, $E_{opt_K}(U^*, V^*) = \alpha N$,
  - If vectors of X are with norm 1, one can choose $V = X$ and $U = \mathbf{I}_N$.

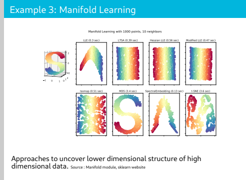
Some comments about the results in the bottom block. If there is no sparsity, and for K higher than the number of dimension, then any basis of the space can be taken and the error is 0. This is a direct consequence of the fact that we are working in a orthonormal space.

Regarding the second item, if taking as many atoms as points in the space, then the error is exactly αN, by simply normalizing vectors of X to norm 1, then choose X as dictionary, and the identity as code.

Example 3: Manifold Learning



Approaches to uncover lower dimensional structure of high dimensional data. Source : Manifold module, sklearn website



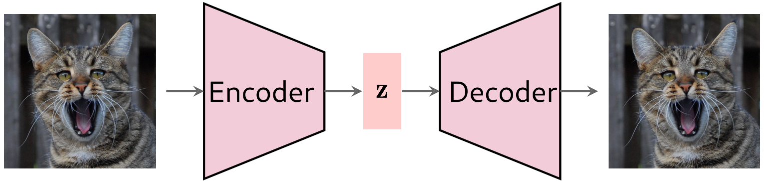
Tell them here that we don't have time to investigate in detail how these different methods work. The important thing is to explain the range of methods that can uncover the lower dimensional topology, in an unsupervised way.

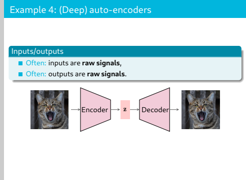
Re-explain the original data (the swiss roll in the top right corner) and explain that there are methods that use different metrics (potentially non linear ones) that try to project in lower d.

Example 4: (Deep) auto-encoders

Inputs/outputs

- Often: inputs are **raw signals**,
- Often: outputs are **raw signals**.



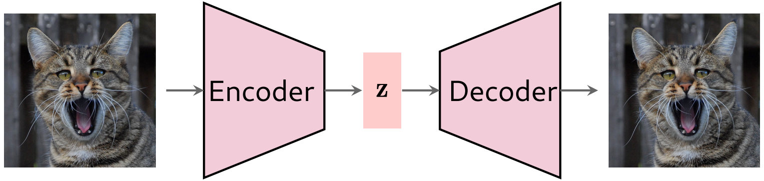


The key point of Autoencoders is the fact that the latent representation (z) is of smaller dimension than the input (generallly largely smaller), hence forcing compression of the input. This means that, to reconstruct the input, the network has to learn a compressed representation of the input, hopefully relying on the most important parts of the image.

Example 4: (Deep) auto-encoders

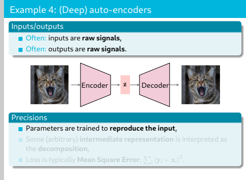
Inputs/outputs

- Often: inputs are **raw signals**,
- Often: outputs are **raw signals**.



Precisions

- Parameters are trained to **reproduce the input**,
- Some (arbitrary) **intermediate representation** is interpreted as the **decomposition**,
- Loss is typically **Mean Square Error**: $\sum_i (\mathbf{y}_i - \mathbf{x}_i)^2$.

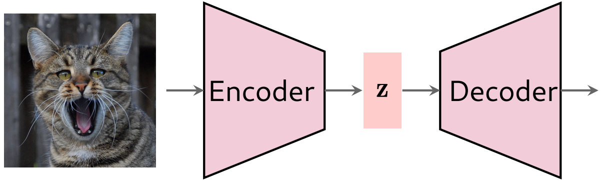


The key point of Autoencoders is the fact that the latent representation (z) is of smaller dimension than the input (generallly largely smaller), hence forcing compression of the input. This means that, to reconstruct the input, the network has to learn a compressed representation of the input, hopefully relying on the most important parts of the image.

Example 4: (Deep) auto-encoders

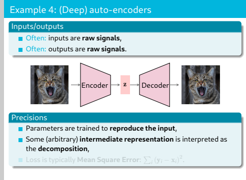
Inputs/outputs

- Often: inputs are **raw signals**,
- Often: outputs are **raw signals**.



Precisions

- Parameters are trained to **reproduce the input**,
- Some (arbitrary) **intermediate representation** is interpreted as the **decomposition**,
- Loss is typically **Mean Square Error**: $\sum_i (\mathbf{y}_i - \mathbf{x}_i)^2$.

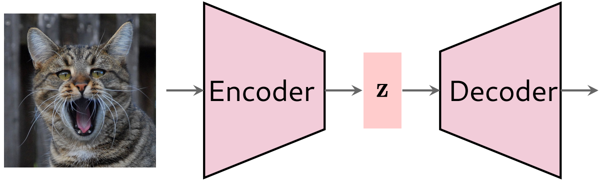


The key point of Autoencoders is the fact that the latent representation (z) is of smaller dimension than the input (generallly largely smaller), hence forcing compression of the input. This means that, to reconstruct the input, the network has to learn a compressed representation of the input, hopefully relying on the most important parts of the image.

Example 4: (Deep) auto-encoders

Inputs/outputs

- Often: inputs are **raw signals**,
- Often: outputs are **raw signals**.

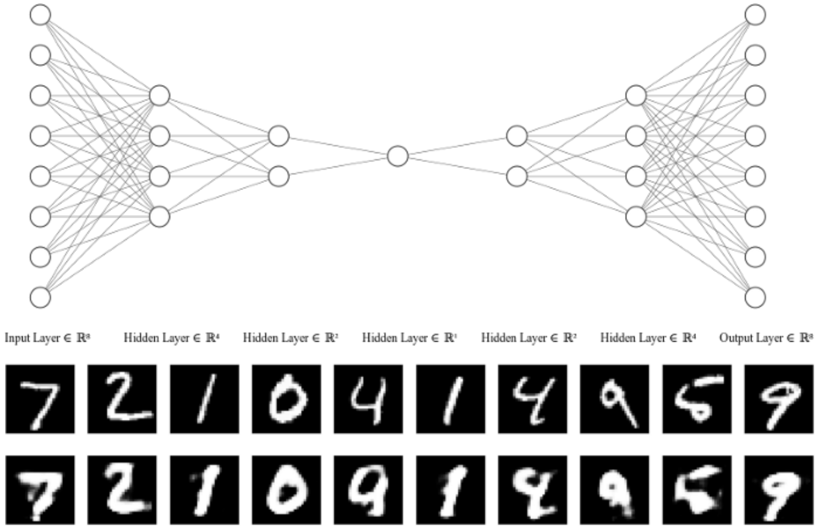


Precisions

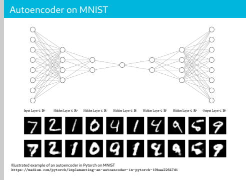
- Parameters are trained to **reproduce the input**,
- Some (arbitrary) **intermediate representation** is interpreted as the **decomposition**,
- Loss is typically **Mean Square Error**: $\sum_i (\mathbf{y}_i - \mathbf{x}_i)^2$.

Example 4: (Deep) auto-encoders
Inputs/outputs
Often: inputs are raw signals,
Often: outputs are raw signals.
Precisions
Parameters are trained to reproduce the input,
Some (arbitrary) intermediate representation is interpreted as the decomposition,
Loss is typically Mean Square Error: $\sum_i (\mathbf{y}_i - \mathbf{x}_i)^2$.

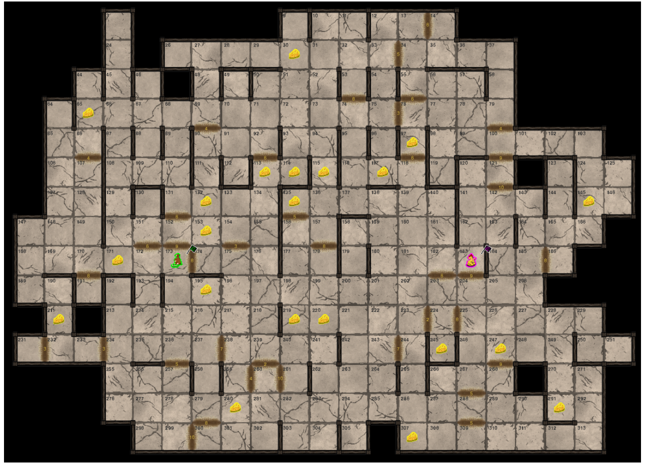
The key point of Autoencoders is the fact that the latent representation (z) is of smaller dimension than the input (generallly largely smaller), hence forcing compression of the input. This means that, to reconstruct the input, the network has to learn a compressed representation of the input, hopefully relying on the most important parts of the image.



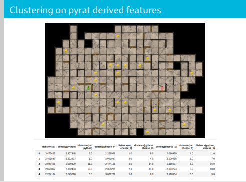
Illustrated example of an autoencoder in Pytorch on MNIST
https://medium.com/pytorch/implementing-an-autoencoder-in-pytorch-19baa22647d1



Particular example on MNIST, with the design of the network (top), followed by the original images (middle) and the reconstructed images (bottom).



	density(rat)	density(python)	distance(rat, python)	density(cheese_0)	distance(rat, cheese_0)	distance(python, cheese_0)	density(cheese_1)	distance(rat, cheese_1)	distance(python, cheese_1)
0	3.475423	2.307948	9.0	2.266966	1.0	8.0	2.016979	4.0	11.0
1	2.401837	2.202623	1.0	2.091537	3.0	4.0	2.109535	6.0	7.0
2	2.948456	2.659309	11.0	2.474181	3.0	14.0	3.116937	5.0	16.0
3	2.655862	2.352433	13.0	2.355235	2.0	11.0	2.183774	3.0	10.0
4	2.204104	2.645298	3.0	3.629737	5.0	8.0	3.910904	6.0	9.0



We just state here the goal for the next lab session.

We give you 1000 initial game configurations (two players, 21 cheese pieces) with the following features (66 in total), computed using the distances as shortest path in the graph:

- Distance between the two players $d(p, r)$
- Distance between each player and each cheese
- Density of cheese around each player starting position $density(p) = \sum_c \frac{1}{d(p,c)}$
- Density of cheese around each cheese position $density(c) = \sum_{c' \neq c} \frac{1}{d(c,c')}$
- Cheese are sorted according to the ratio $o(c) = \frac{d(r,c)}{d(p,c)}$

Your task : Find clusters in this dataset, we will evaluate your cluster labels using the ground truth.
more details in the lab session 3 notebook

Clustering on pyrat derived features

We give you 1000 initial game configurations (two players, 21 cheese pieces) with the following features (66 in total), computed using the distances as shortest path in the graph:

- Distance between the two players $d(p, r)$
- Distance between each player and each cheese
- Density of cheese around each player starting position $density(p) = \sum_c \frac{1}{d(p,c)}$
- Density of cheese around each cheese position $density(c) = \sum_{c' \neq c} \frac{1}{d(c,c')}$
- Cheese are sorted according to the ratio $o(c) = \frac{d(r,c)}{d(p,c)}$

Your task : Find clusters in this dataset, we will evaluate your cluster labels using the ground truth.
more details in the lab session 3 notebook

We just state here the goal for the next lab session.

N.b. : valid in unsupervised and supervised settings.

Feature preprocessing

Objective : change the statistical distribution of the features

- Scaling / Normalization
- Power transform
- Encode, discretization
- Manual feature engineering
- See more `https://scikit-learn.org/stable/modules/preprocessing.html`

Many techniques need or are greatly helped when features are on the unit sphere.

Working with features

N.b. : valid in unsupervised and supervised settings.

Feature preprocessing

Objective : change the statistical distribution of the features

- Scaling / Normalization
- Power transform
- Encode, discretization
- Manual feature engineering
- See more https://scikit-learn.org/stable/modules/preprocessing.html

Many techniques need or are greatly helped when features are on the unit sphere.

Don't hesitate to state that this lab is not easy, and that we value exploration and justification of the tests over results.

Working with features

N.b. : valid in unsupervised and supervised settings.

Feature selection

Objective : remove features

- Remove features with low variance
- Select features according to their explained variance towards labels (e.g. SelectKBest)
- See more `https://scikit-learn.org/stable/modules/feature_selection.html`

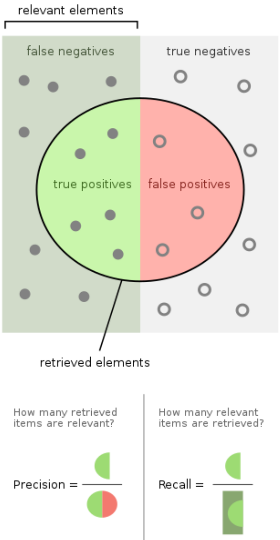
Helps to adress the dimensionality curse.

Working with features
N.b. : valid in unsupervised and supervised settings.
Feature selection
Objective : remove features
Remove features with low variance
Select features according to their explained variance towards labels (e.g. SelectKBest)
See more https://scikit-learn.org/stable/modules/feature_selection.html
Helps to adress the dimensionality curse.

Don't hesitate to state that this lab is not easy, and that we value exploration and justification of the tests over results.

Metrics

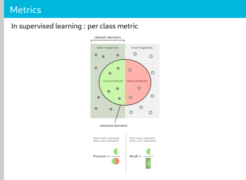
In supervised learning : per class metric



2024-02-13

Course 3: Unsupervised Learning

└Metrics



Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics
Clustering Metrics :
Error defined slide 5 : similar to inertia (sum of squared distances)
The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).
Clustering metrics using labels :
Random Index : measures the similarity of two assignments, ignoring permutations
See more on sklearn website and in the lab session

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Metrics

Clustering Metrics :

- Error defined slide 5 : similar to inertia (sum of squared distances)
- The Silhouette score (per sample) is $(b - a)/max(a, b)$, with mean intra-cluster distance (a) and the mean nearest-cluster distance (b).

Clustering metrics using labels :

- Random Index : measures the similarity of two assignments, ignoring permutations
- Homogeneity : each cluster contains only members of a single class.
- Completeness : all members of a given class are assigned to the same cluster.

See more on sklearn website and in the lab session

Lab Session 3 and assignments for Session 5

TP Unsupervised Learning (TP2)

- K-means, Dictionary Learning and Manifold Learning
- Application on Digits and PyRat

Project 2 (P2)

- Find clusters in the provided dataset of pyrat games features.
- You can combine every technique you want (feature selection, decomposition, clustering, ...)
- During Session 4 you will have 7 minutes to present your work.
- We will evaluate the quality of your clustering during your presentation.

Lab Session 3 and assignments for Session 5
TP Unsupervised Learning (TP2)
- K-means, Dictionary Learning and Manifold Learning
- Application on Digits and PyRat
Project 2 (P2)
- Find clusters in the provided dataset of pyrat games features.
- You can combine every technique you want (feature selection, decomposition, clustering, ...)
- During Session 4 you will have 7 minutes to present your work.
- We will evaluate the quality of your clustering during your presentation.

Self explanatory!