CE9010: Introduction to Data Science

Lecture 8: Unsupervised Learning

Semester 2 2017/18

Xavier Bresson

School of Computer Science and Engineering
Data Science and AI Research Centre
Nanyang Technological University (NTU), Singapore

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Outline

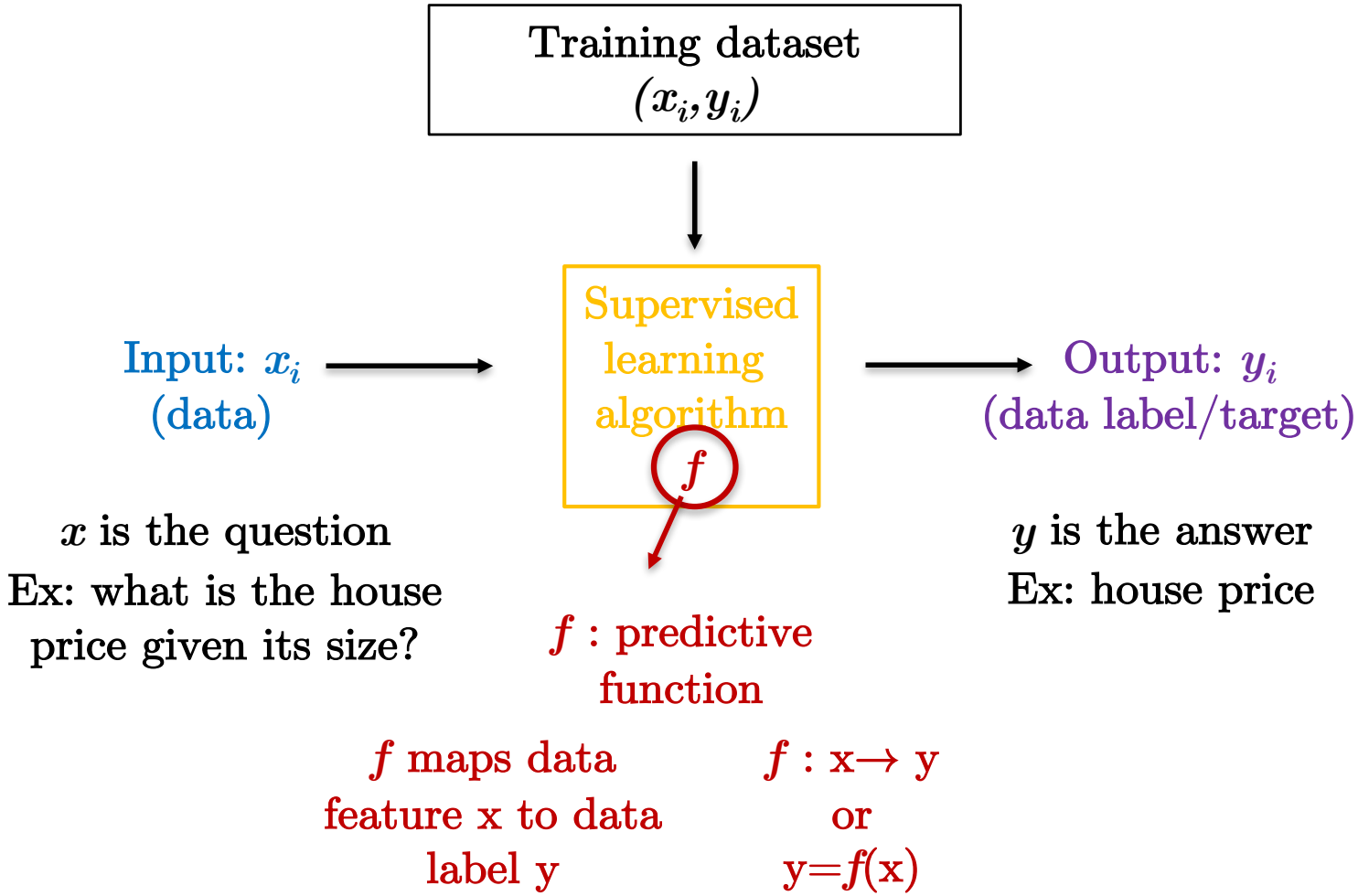
- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Outline

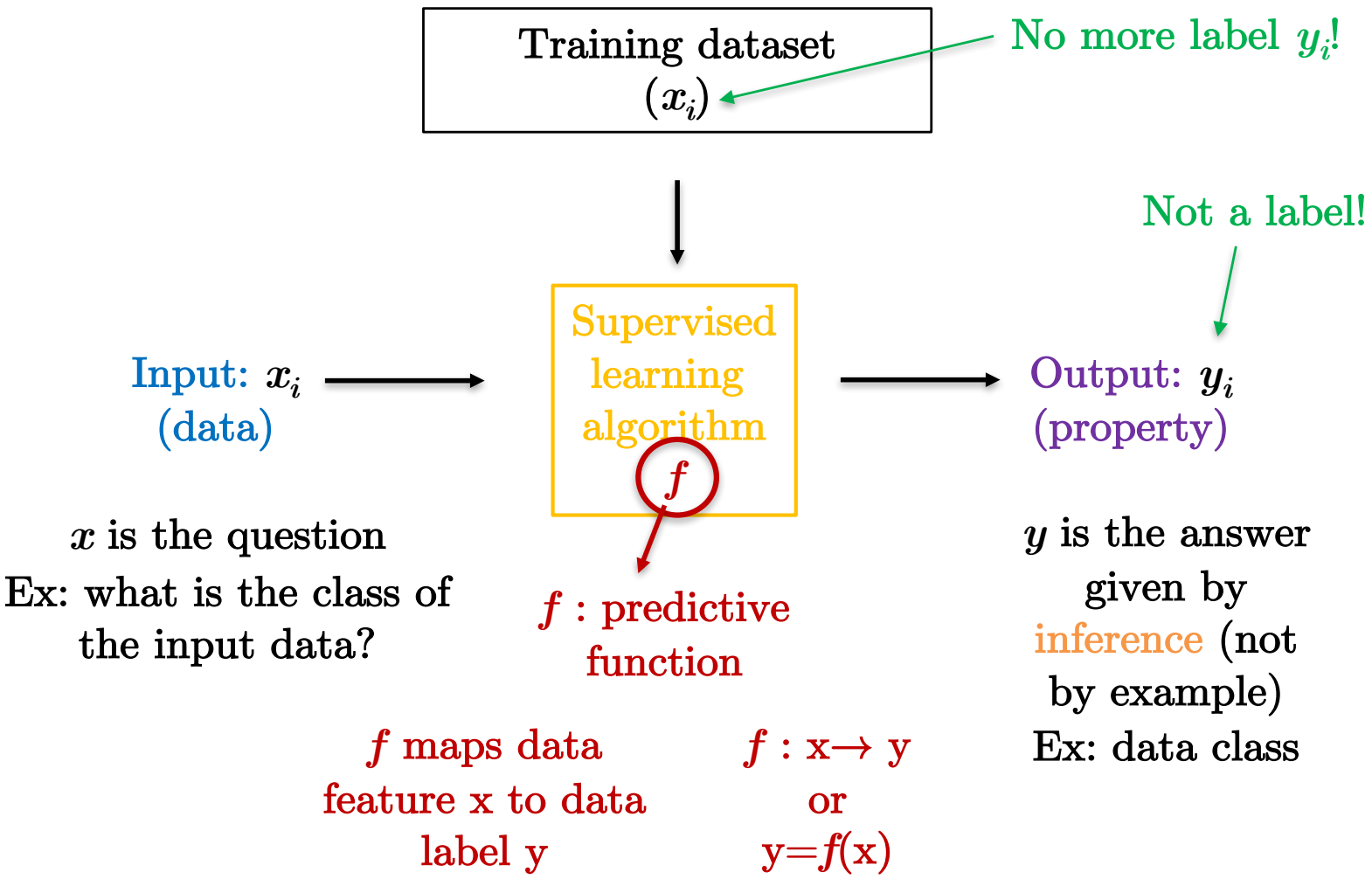
- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Supervised learning

- Supervised learning: Design a predictive function given a training set composed of data points (x_i, y_i) with x_i the data features and y_i the data label.

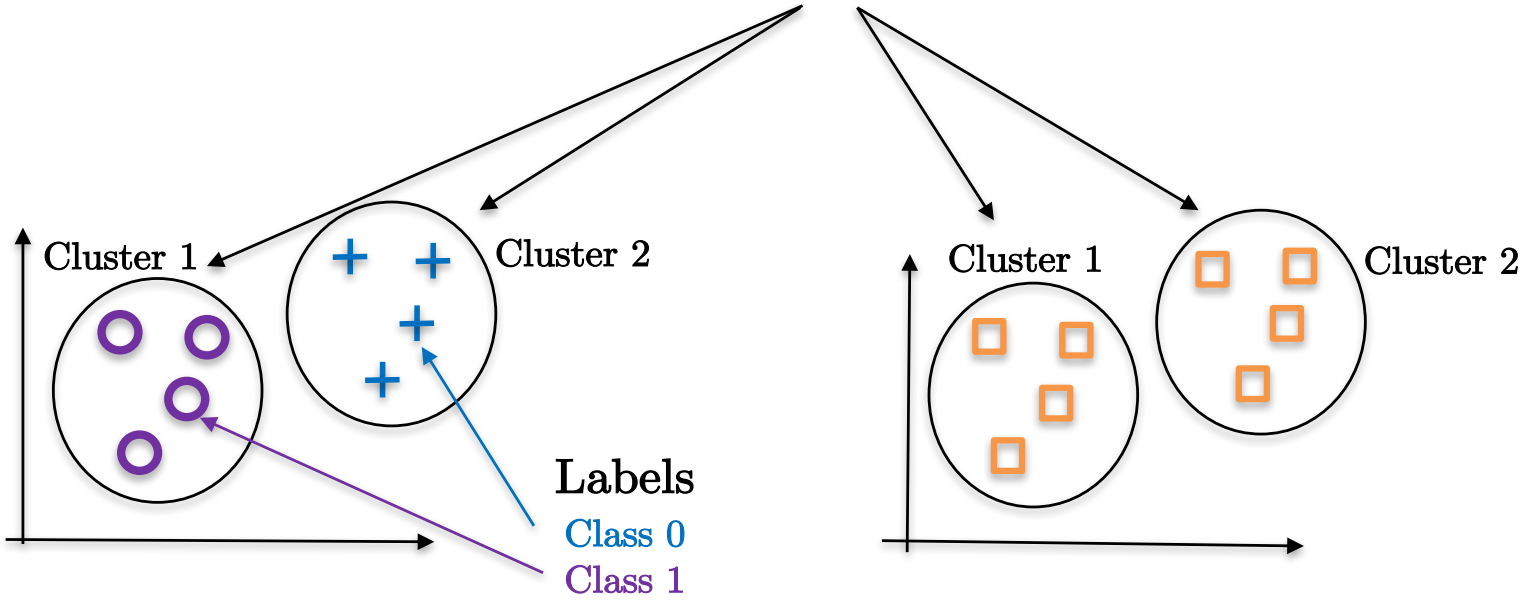


Unsupervised learning



Supervised vs unsupervised

- Difference between supervised and unsupervised learning:
  - Consider the task of finding clusters of data:



Supervised learning

{ data, label }

Labels are highly valuable to solve the learning task.

Unupervised learning

{ data , - }

No labels

Unsupervised learning

- Unsupervised learning: Design a predictive function given a training set which has no label. The training set is simply composed of the data points (x_i) with the data features (no y_i).

- Main idea: Find structures in data that can solve data analysis tasks.

- Common tasks:

  - Unsupervised clustering: k-means (structures are the means)

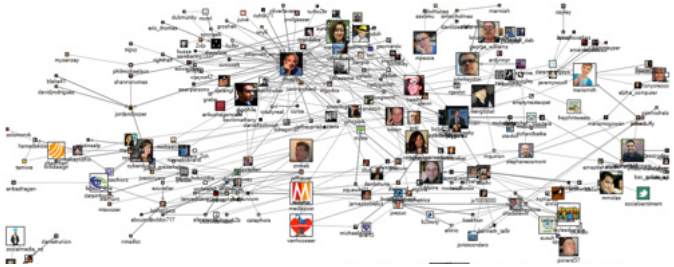
  - Unsupervised representation: PCA (structures are the variances)

Outline

- Supervised vs unsupervised learning
- **Unsupervised clustering with k-means**
  - **Clustering**
  - k-means algorithm
  - Loss
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Unsupervised clustering

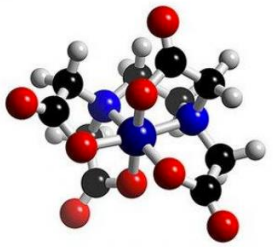
- Definition: Find groups of data that are similar.
- Applications:



Social network analysis
(ads for groups of similar people)

Market analysis
(segment customers or products in homogeneous groups)

Image segmentation
(group pixels of same objects)



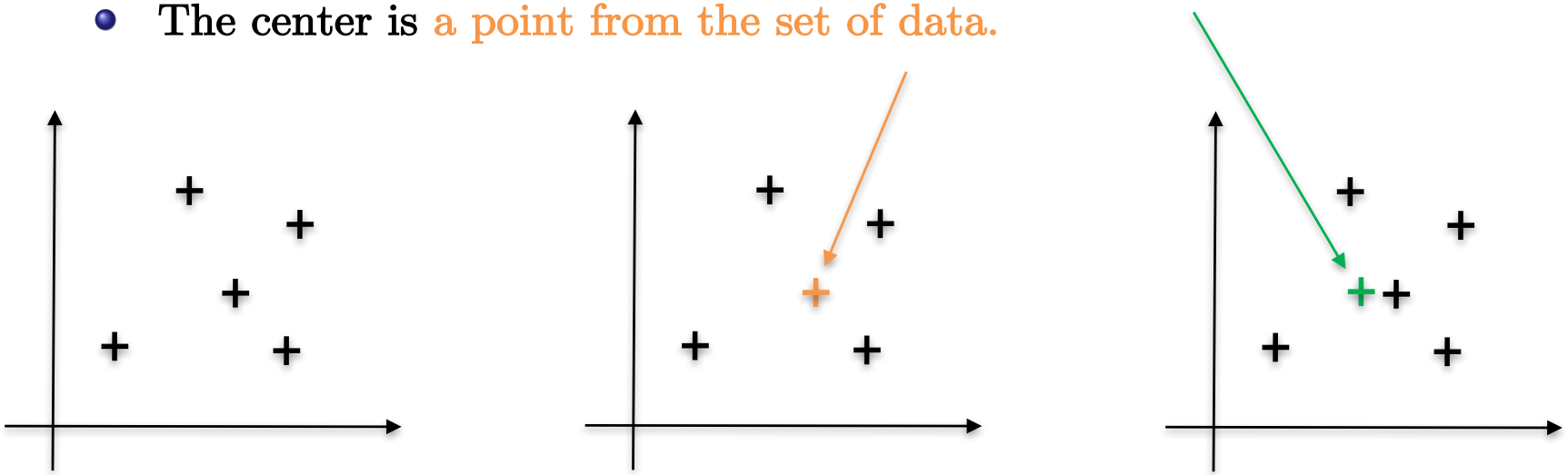
Molecular analysis
(find molecules related to drugs)

Outline

- Supervised vs unsupervised learning
- **Unsupervised clustering with k-means**
  - Clustering
  - **k-means algorithm**
  - Loss
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

k-means

- k-means algorithm is the most popular unsupervised clustering algorithm.
- Definition: The mean (a.k.a. cluster center, centroid) of a cluster is the center of this cluster.
- Center definition:
  - The center is a point not necessarily belonging to the set of data.
  - The center is a point from the set of data.

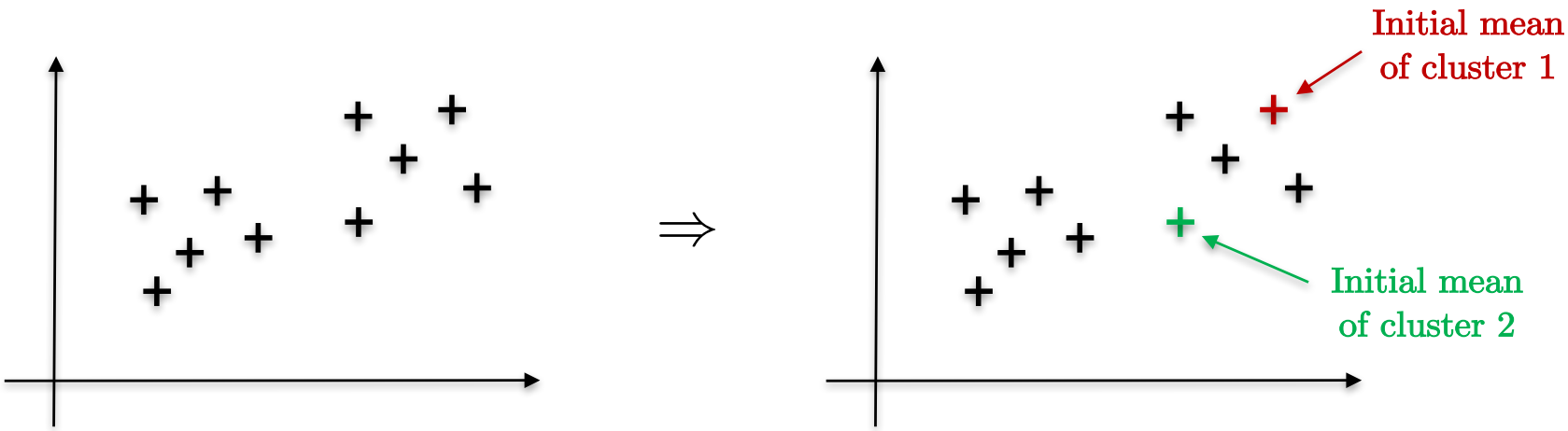


Data points

- In this lecture, we choose the center to be a point from the set of data.

k-means

- k-means algorithm steps:
  - Step 1: Initialization. Select randomly k points called means:

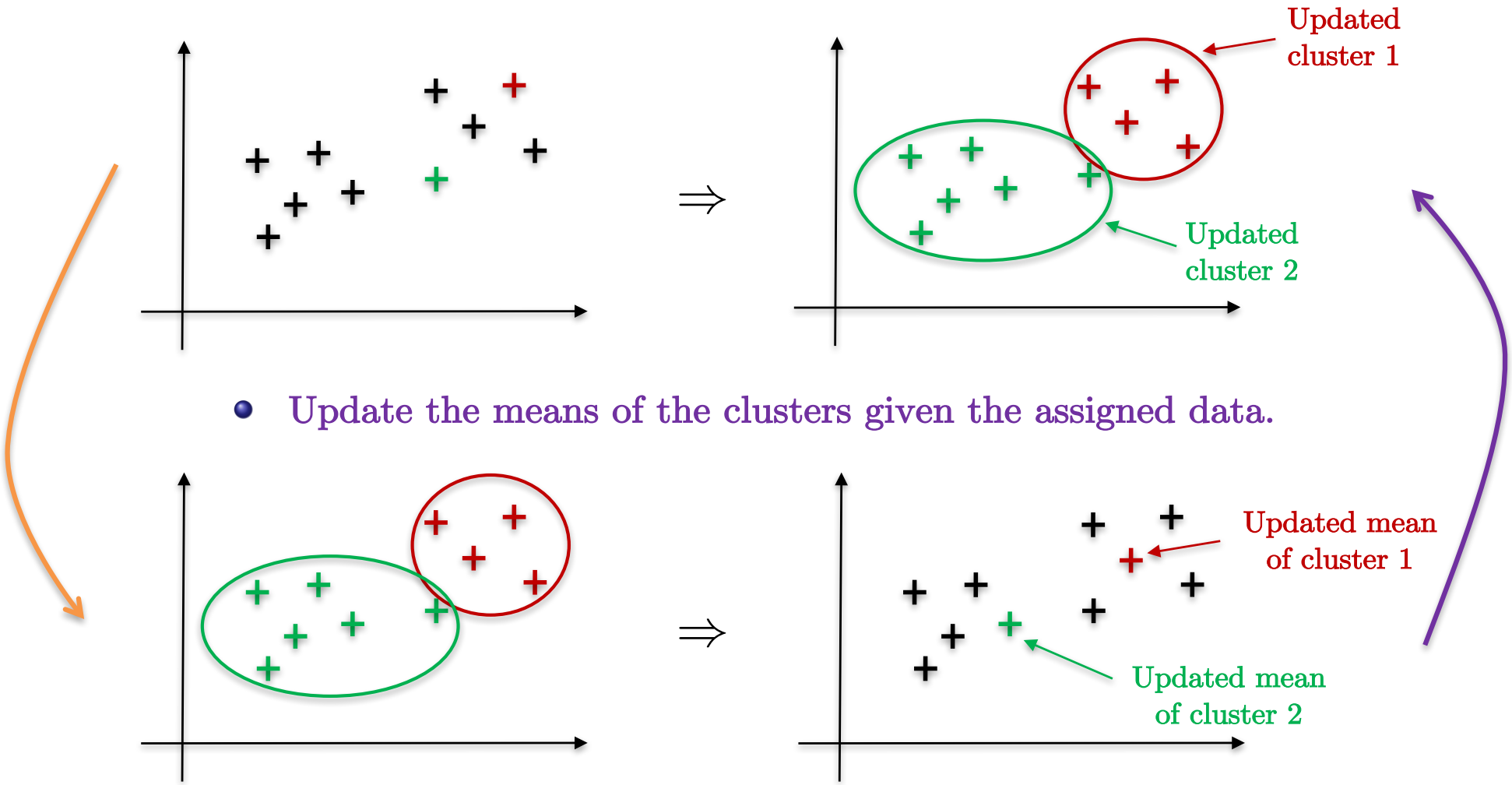


Original data points

Step 1: Initialization

k-means

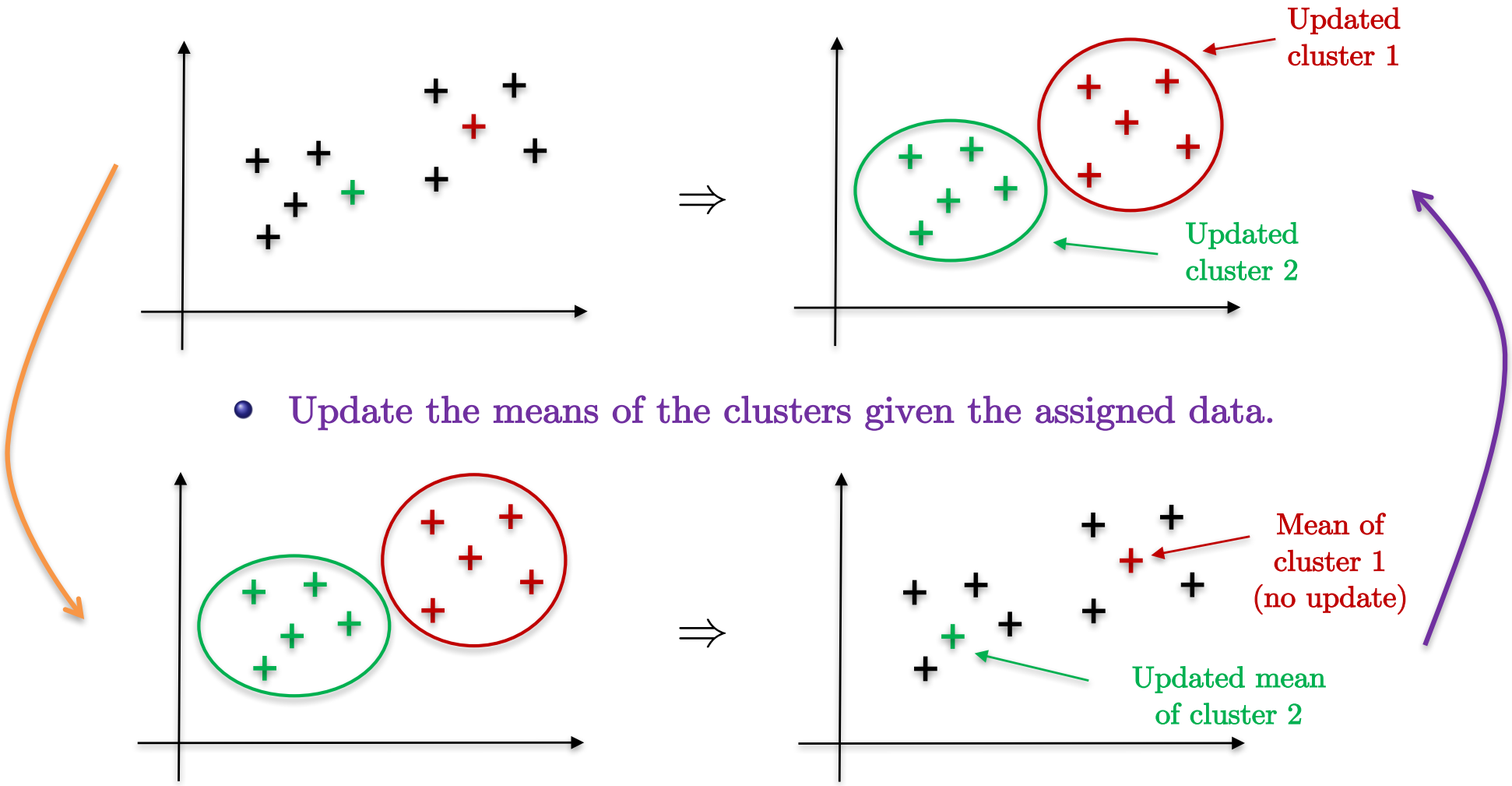
- Step 2: Loop until convergence.
  - Assign each data to one of the 2 means.



  - Update the means of the clusters given the assigned data.

k-means

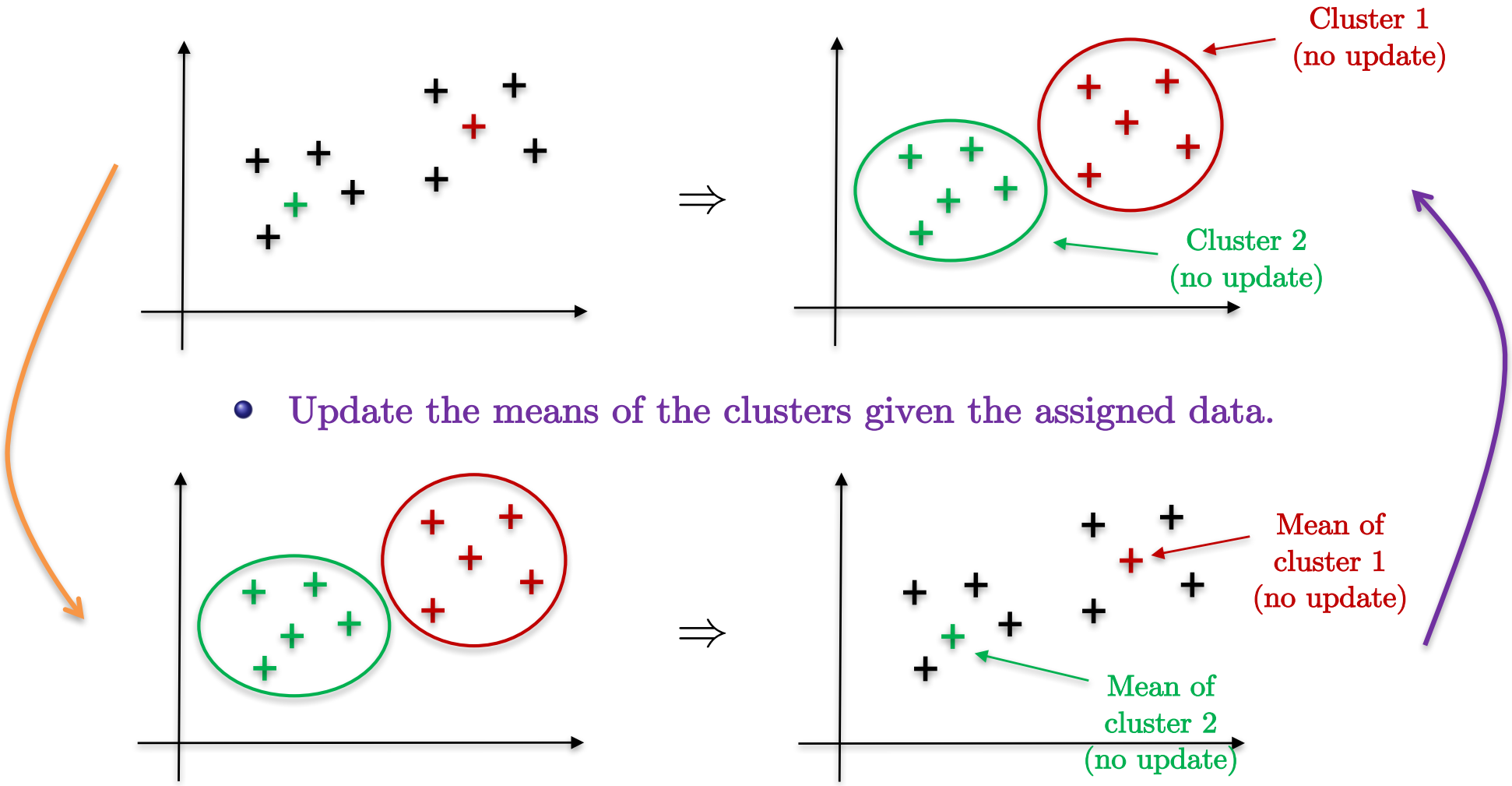
- Step 2: Loop until convergence.
  - Assign each data to one of the 2 means.



  - Update the means of the clusters given the assigned data.

k-means

- Step 2: Loop until convergence.
  - Assign each data to one of the 2 means.



  - Update the means of the clusters given the assigned data.

Formalization

- Inputs:
  - K = number of clusters
  - Training set $\{x_1,\ldots,x_n\}$, x_i *in* R^d, d=number of features/dim
- Outputs:
  - Clusters = $\{C_1,\ldots,C_K\}$ and means = $\{\ \mu_1,\ldots,\ \mu_K\ \}$.
- K-means algorithm:
  - Initialization: Randomly initialize K means: $\{\ \mu_1,\ldots,\ \mu_K\ \} \in \mathrm{R}^{\mathrm{d}}$.
  - Repeat until convergence:
    1. Clusters assignment $\{C_1,\ldots,C_K\}$
    2. Means update $\{\ \mu_1,\ldots,\ \mu_K\ \}$.

Formalization

1. Cluster assigmnent

For all $i = 1$ to n:

a_i gives the index of the cluster the closest to x_i.

$$a_i = \arg\min_k \|x_i - \mu_k\|_2^2$$

Take the k that minimizes the distance

Distance between x_i and the mean μ_k.

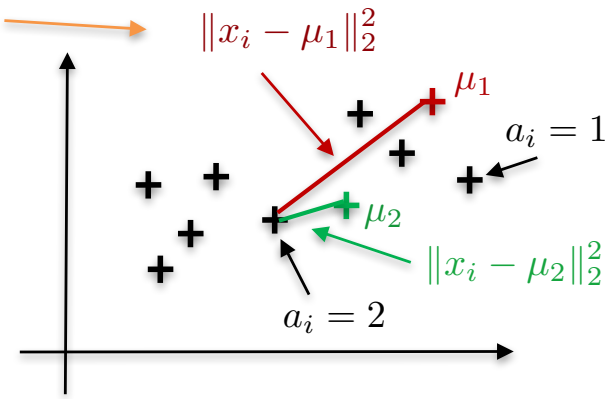
with $\|x\|_2^2 = \sum_{j=1}^{d} x_j^2$

Euclidean distance

$$a = \begin{bmatrix} 2 \\ 1 \\ 1 \\ \vdots \\ 2 \end{bmatrix} \quad n \times 1$$

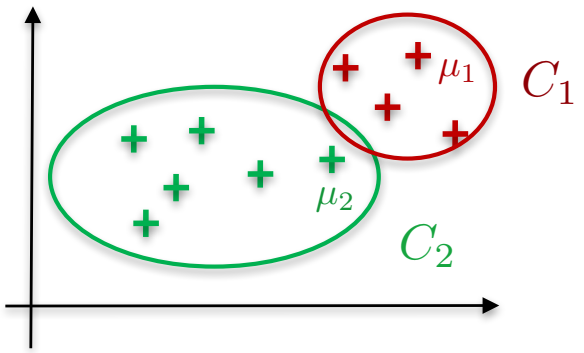
1st data in C_2

3rd data in C_1



For all $k = 1$ to K:

$$C_k = \{x_i : a_i = k\}$$



Formalization

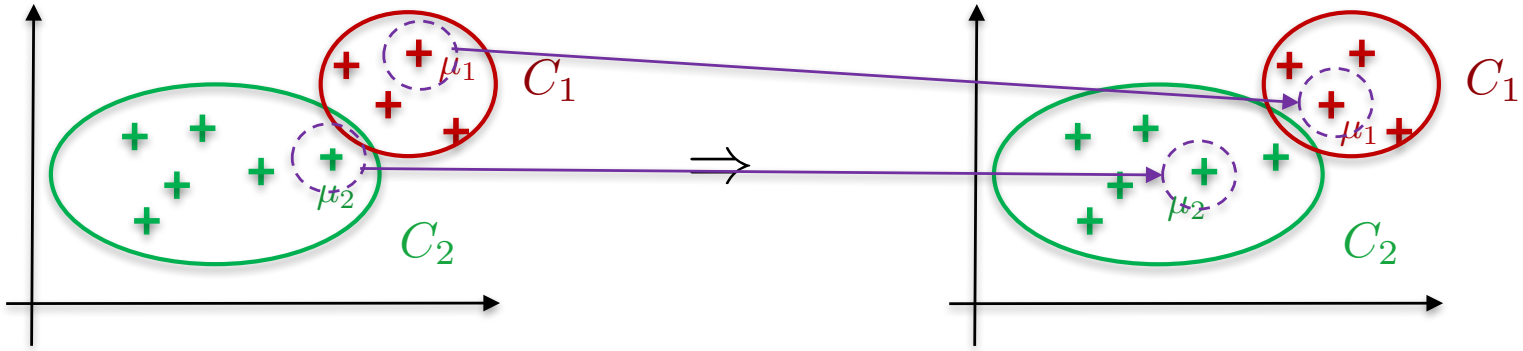
2. Mean update

For all $k = 1$ to K:

$$\underset{d \times 1}{\mu_k} = \frac{\sum_{x_i \in C_k} x_i}{|C_k|}$$

μ_k is the mean of all data points in cluster C_k

nb of data in cluster C_k



Outline

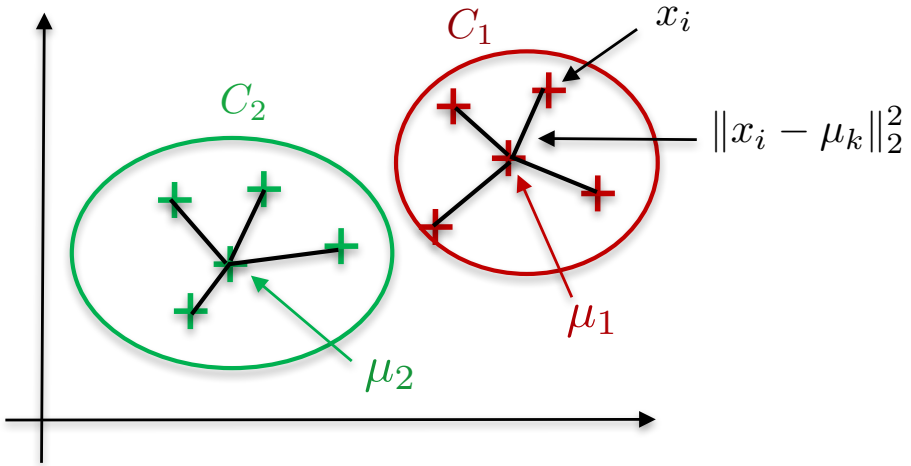
- Supervised vs unsupervised learning
- **Unsupervised clustering with k-means**
  - Clustering
  - k-means algorithm
  - **Loss**
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

k-means loss

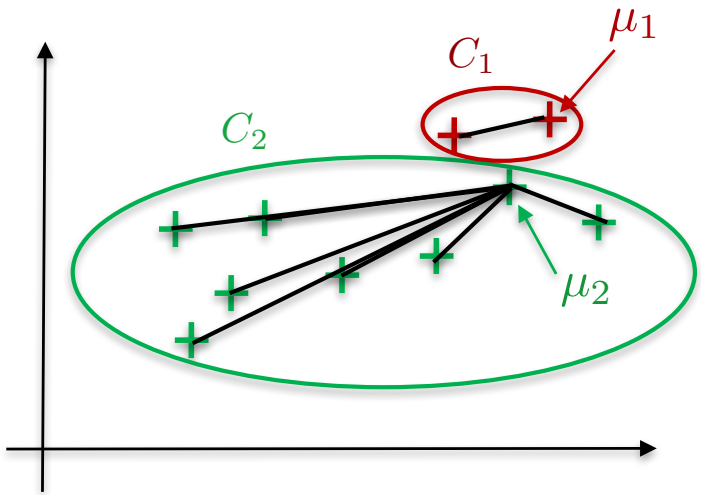
- Measure of fitness/accuracy between the k-means model and the data points:

$$L(\underbrace{C_1, ..., C_K}_{\text{Clusters}}, \underbrace{\mu_1, ..., \mu_K}_{\text{Means}}) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - \mu_{a_i}\|_2^2$$

$$= \frac{1}{n}\sum_{k=1}^{K}\sum_{x_i \in C_k} \|x_i - \mu_k\|_2^2$$



Loss is small (minimized)



Loss is large

Re-visiting k-means algorithm

- Optimize the k-means loss:

$$\min_{\substack{C_1,...,C_K \\ \mu_1,...,\mu_K}} L(C_1,...,C_K,\mu_1,...,\mu_K)$$

Find C_k and μ_k that minimize the loss function.

Re-visiting k-means algorithm

- **Repeat:**

  1. Cluster assignment

$$\min_{C_1,...,C_K} L(C_1, ..., C_K) \quad \text{for } \mu_1, ..., \mu_K \text{ fixed}$$

$$\Downarrow$$

$$C_k = \{x_i : a_i = k\}$$

$$a_i = \arg\min_k \|x_i - \mu_k\|_2^2$$

  2. Mean update

$$\min_{\mu_1,...,\mu_K} L(\mu_1, ..., \mu_K) \quad \text{for } C_1, ..., C_K \text{ fixed}$$

$$\Downarrow$$

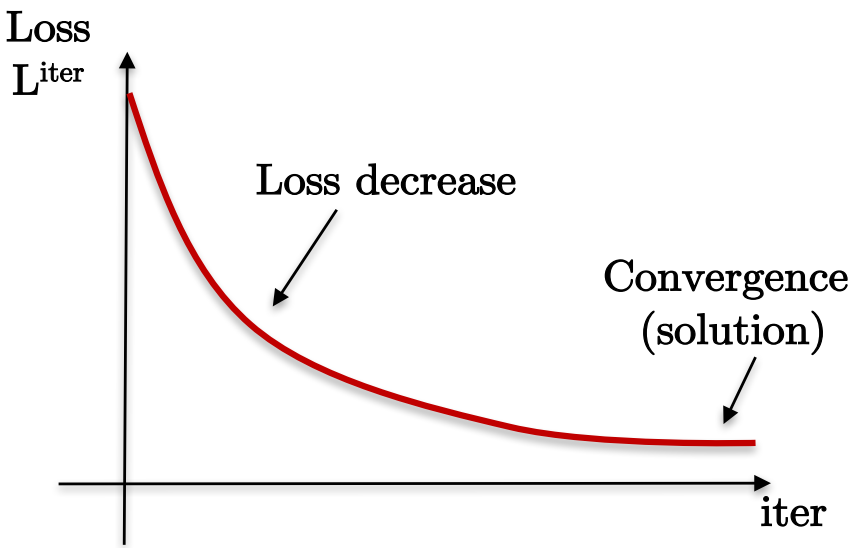
$$\mu_k = \frac{\sum_{x_i \in C_k} x_i}{|C_k|}$$

Outline

- Supervised vs unsupervised learning
- **Unsupervised clustering with k-means**
  - Clustering
  - k-means algorithm
  - Loss
  - **k-means properties**
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Monotonicity and non-convexity

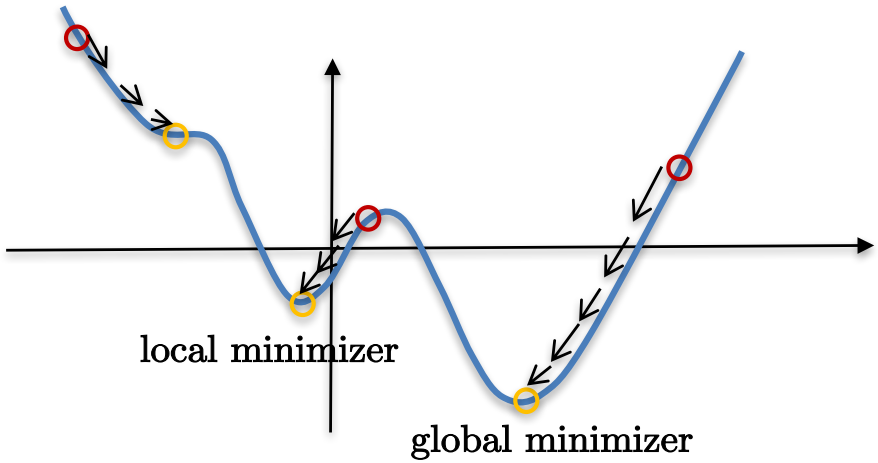
- Monotonicity: k-means is guaranteed to decrease the loss at each iteration.



- Non-convex: k-means loss is non-convex

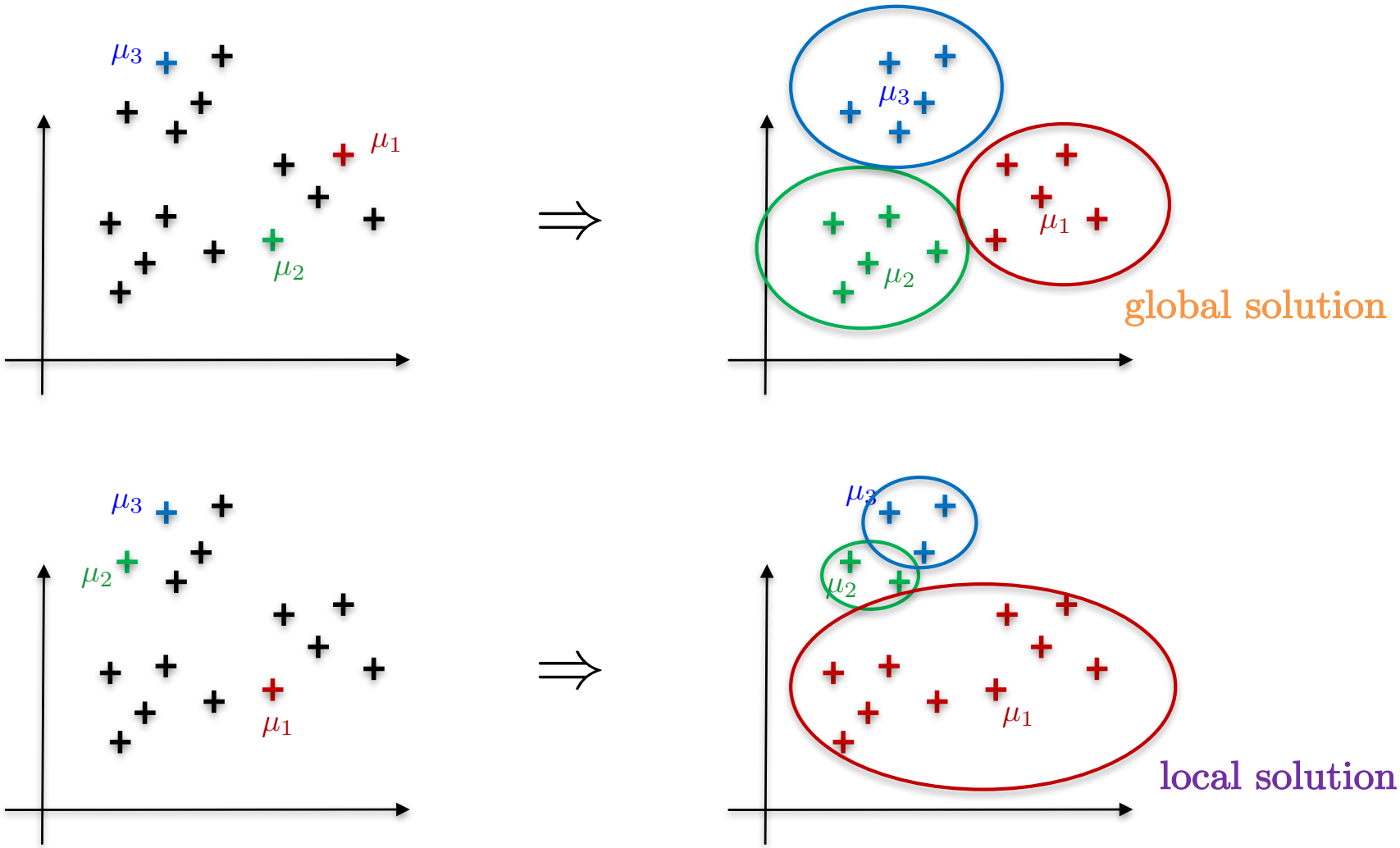
  ⇒ No guarantee to find the global minimizer of the loss.

  ⇒ Initial condition is important.



Initialization

- k-means has no guarantee to find a global solution:

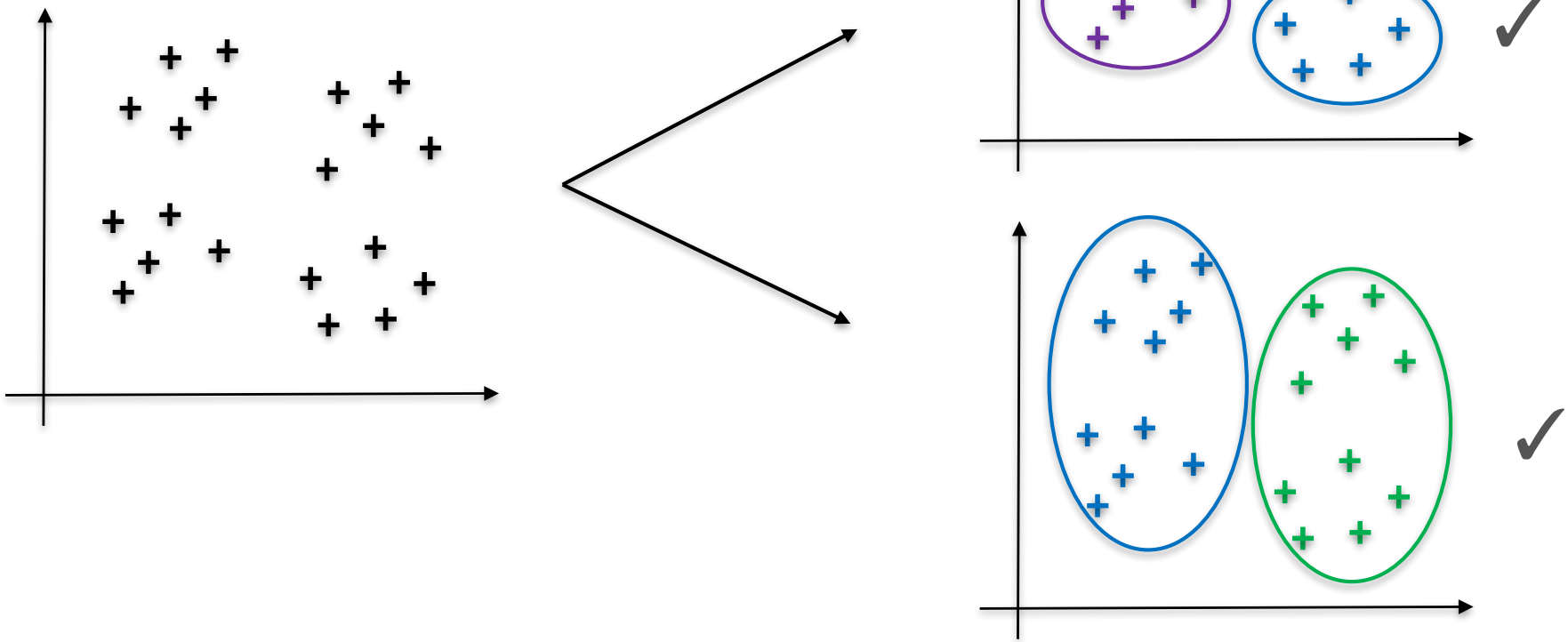


Initialization

- Reduce the dependence w.r.t. initialization:
  - Run the k-means algorithm 100 times with random initialization.
  - Save clusters and the loss value for each run.
  - Pick the solution with the smallest loss value.

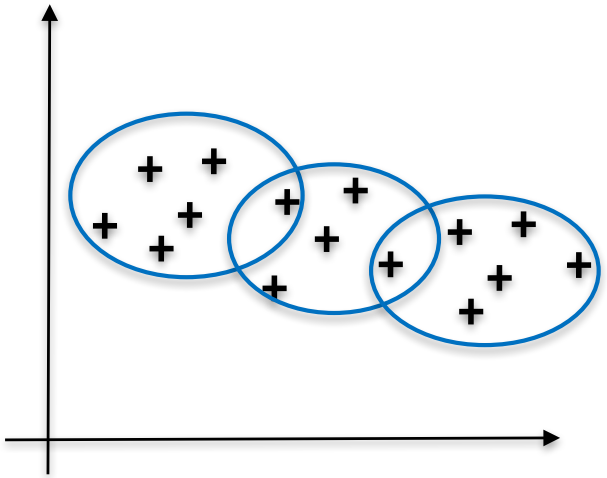
How to choose the number k of clusters?

- No obvious answer. It depends on the data and the next task that will make use of the clusters (sometimes, post-processing selection after k-means ran).
- Example: Ambiguous "optimal" choice of k?

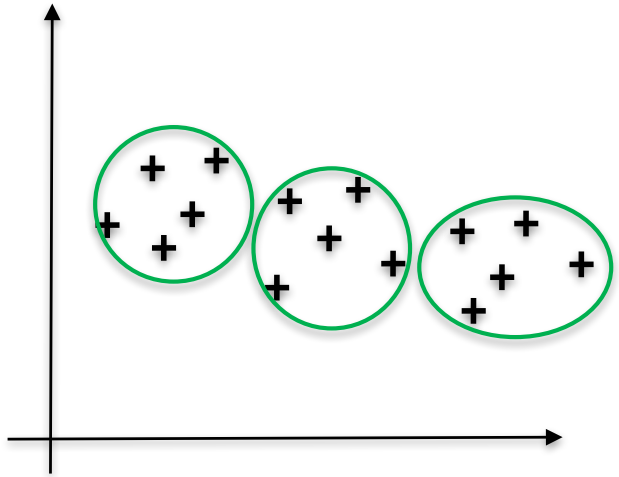


Separated clusters

- K-means algorithm cannot be applied to non-separated/ overlapping clusters. It is designed to find separated clusters.



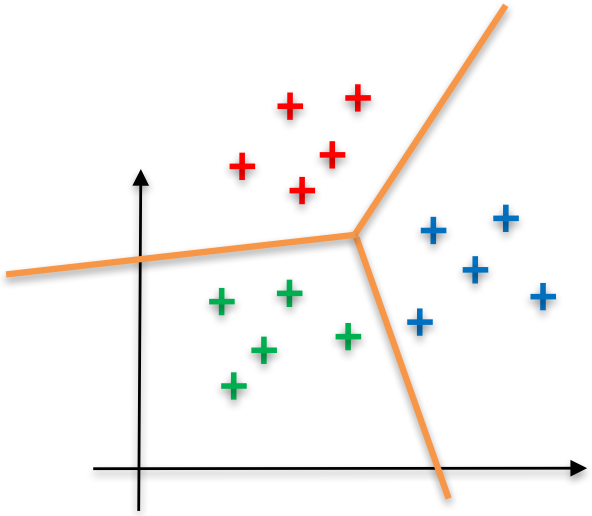
Overlapping clusters



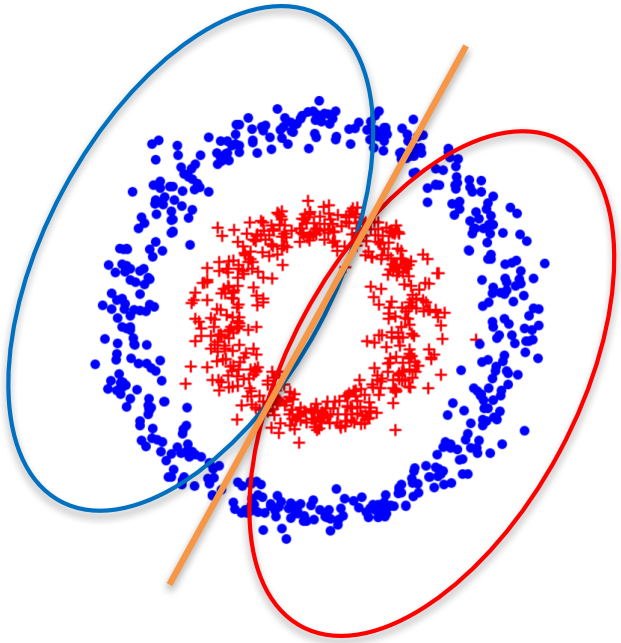
Separated clusters

Limitation

- k-means algorithm assumes the data are linearly separable ⇒ Data can be separated by straight lines.



- k-means is not supposed to work for data with more complex distributions (solution is given by kernel k-means).



Summary

- k-means is the oldest unsupervised clustering algorithm (by physicist Lloyd in 1957).
- k-means is sound: Loss decrease and convergence guaranteed, speed is linear w.r.t. the number of data, the number of features and the number of clusters, easy to implement, and multiple extensions exist (k-medians, k-plans)
- k-means is limited: linear data and existence of local minimizers.

Outline

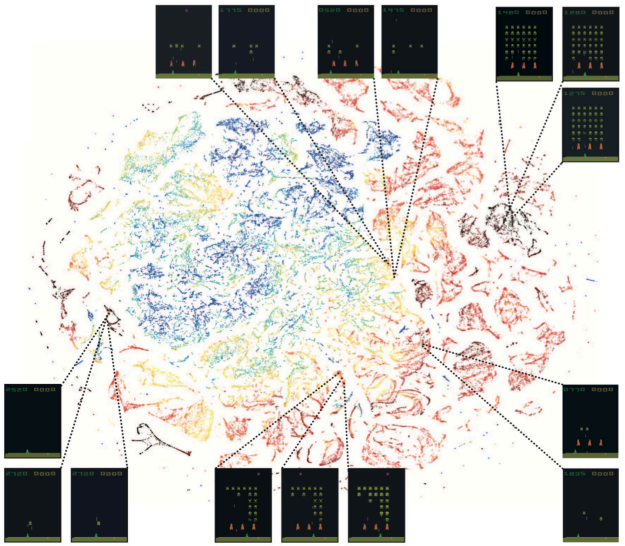
- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- **Unsupervised representation with PCA**
  - **Introduction and motivations**
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Dimensionality reduction

- Principal component analysis (PCA) is a dimensionality reduction technique (introduced by statistician Karl Pearson in 1901).

- Why dimensionality reduction is useful?
  - Data compression (compact representation)
  - Data visualization (2D and 3D)

Data compression
64x64≈4k reduced to
10 variables

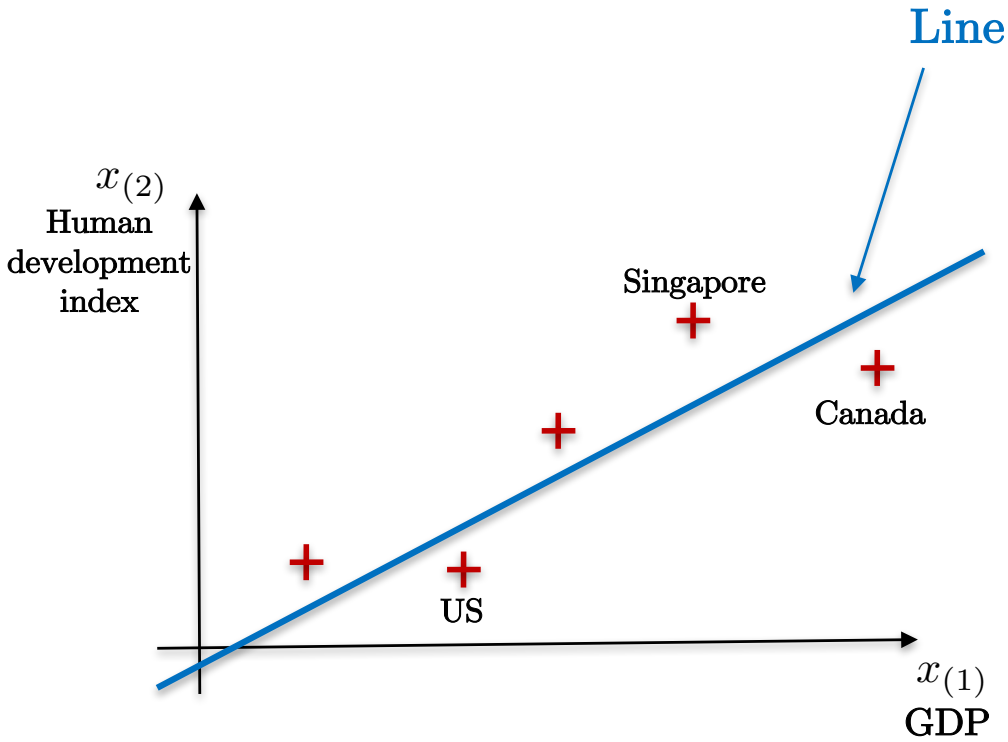


Data visualization
Video games

Data compression motivation

- Example: Let's consider some country properties

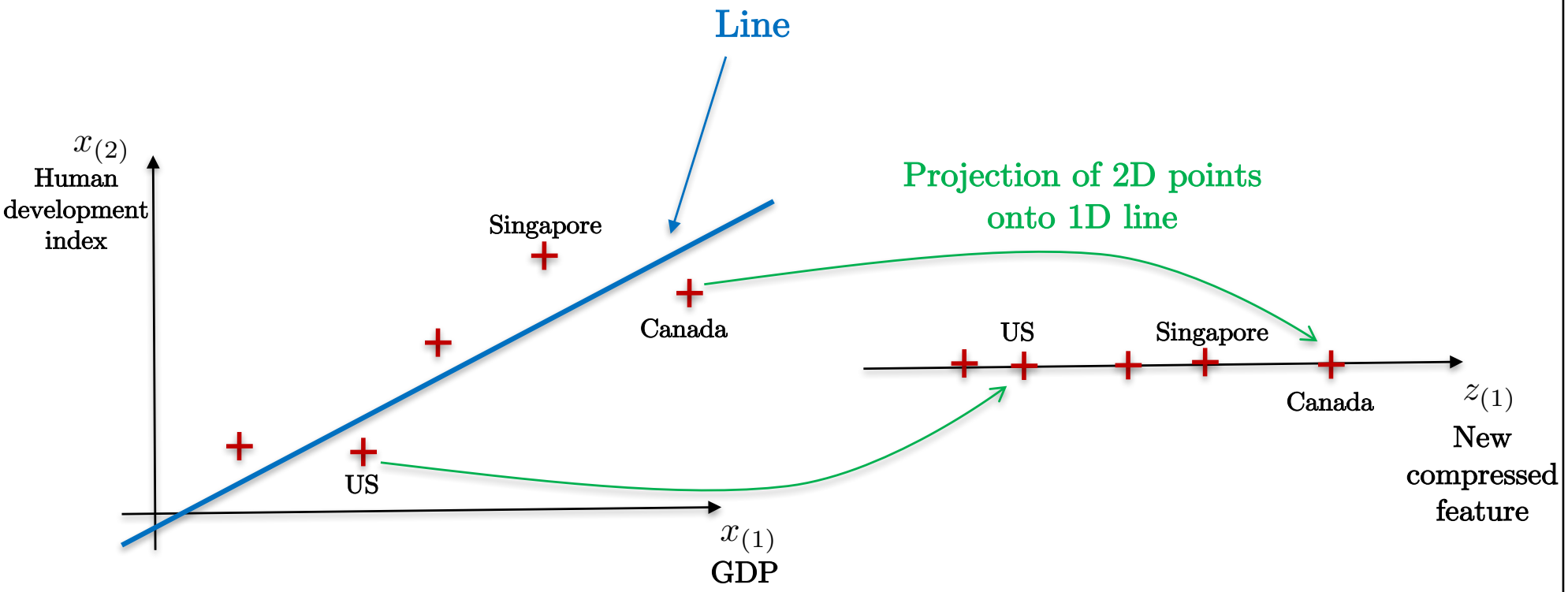
Country	GDP (gross domestic product in trillions of US$) $x_{(1)}$	Human development index $x_{(2)}$
US	14.3	0.91
Canada	1.5	0.90
China	5.8	0.68
India	1.6	0.5
Singapore	0.2	0.8
Russia	1.4	0.7



Data are close to a line. There is some redundancy between the 2 features representing the country ⇒ Compression is possible.

Data compression motivation

- We can reduce the data from 2 features (2D representation) to 1 feature (1D representation):



Compression

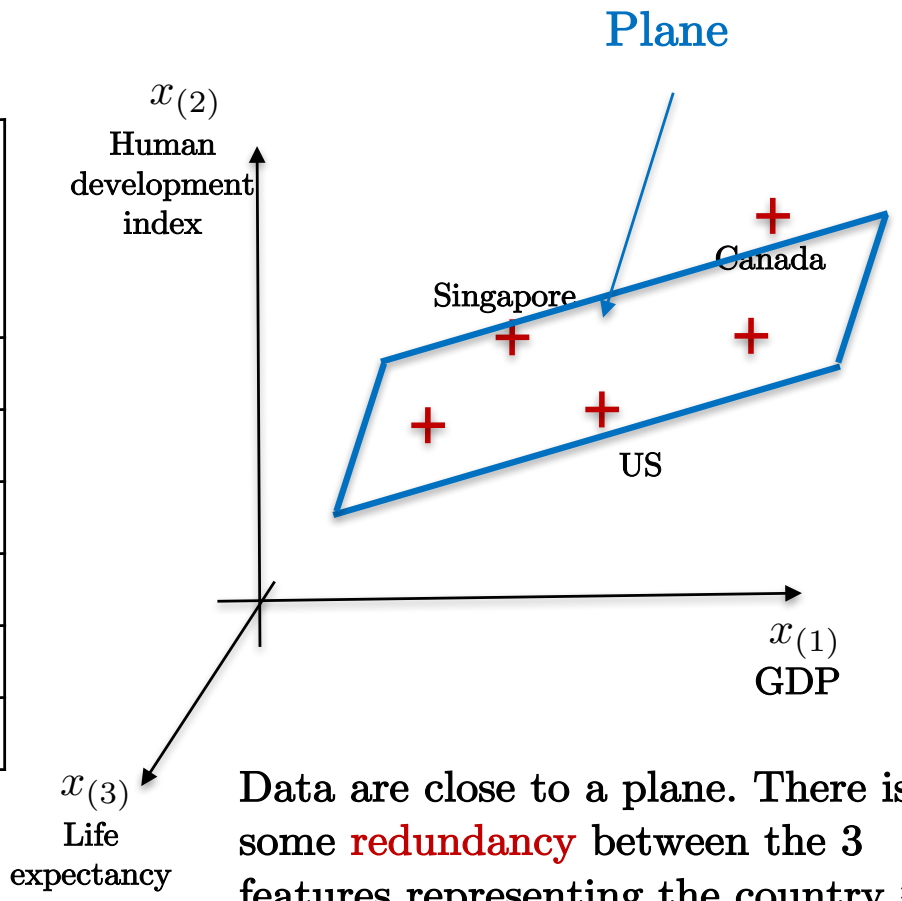
2D representation $\Rightarrow$ 1D representation

$x \in \mathbb{R}^2$ $\quad$ $z \in \mathbb{R}^1$

Data compression motivation

- Example: Let's consider three country properties

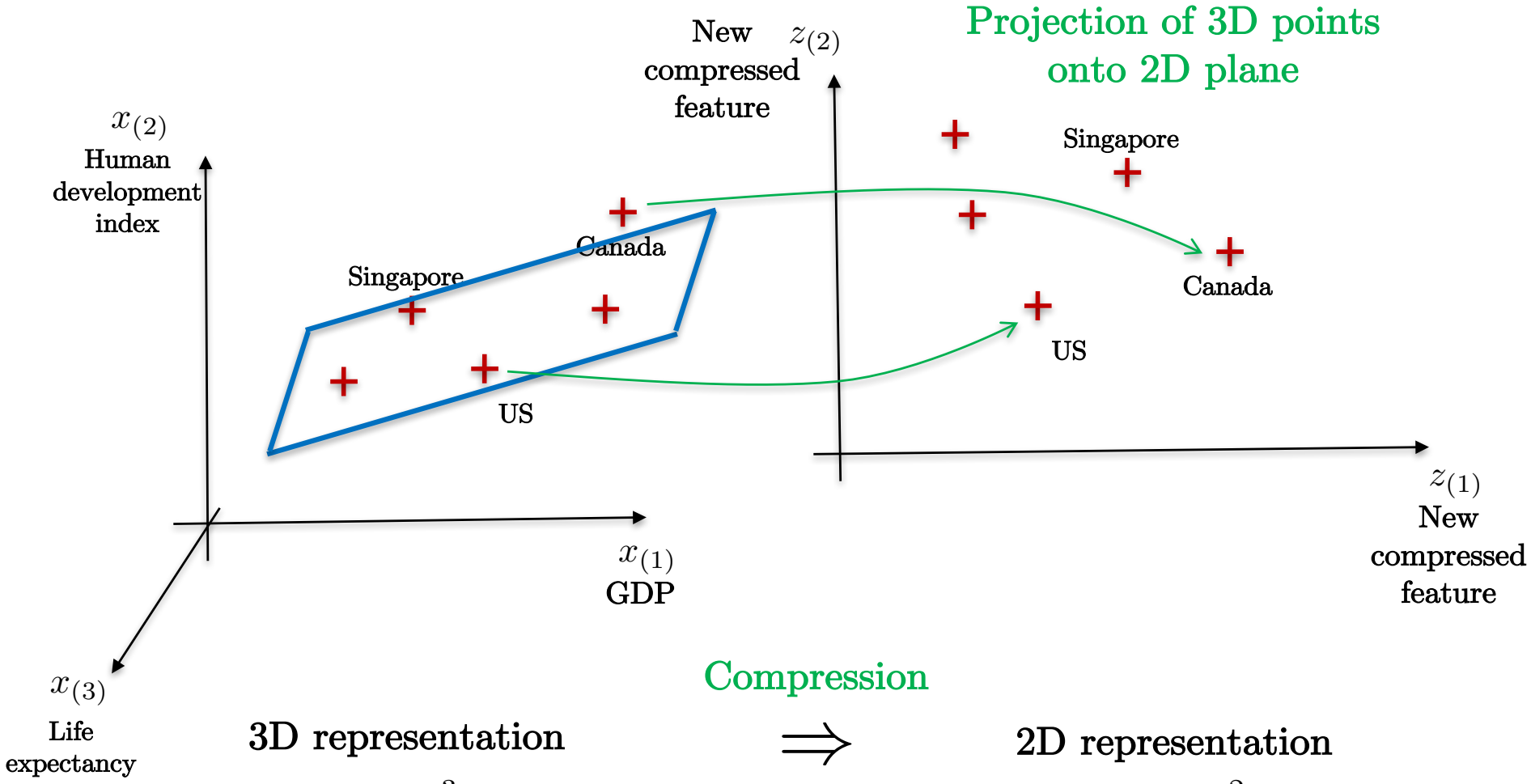
Country	GDP (gross domestic product in trillions of US$) $x_{(1)}$	Human development index $x_{(2)}$	Life expectancy $x_{(3)}$
US	14.3	0.91	78.3
Canada	1.5	0.90	80.7
China	5.8	0.68	73
India	1.6	0.5	64.7
Singapore	0.2	0.8	80
Russia	1.4	0.7	65.5



Data are close to a plane. There is some redundancy between the 3 features representing the country ⇒ Compression is again possible.

Data compression motivation

- We can also reduce the data from 3 features (3D representation) to 2 features (2D representation):



Data compression motivation

- General case: Data features can be compressed from 10,000-D compressed to 100-D.

Dim reduction/
Compression

10,0000-D representation $\Longrightarrow$ 100-D representation

$$x = \begin{bmatrix} x_{(1)} \\ x_{(2)} \\ \vdots \\ x_{(10,000)} \end{bmatrix} \in \mathbb{R}^{10,000} \qquad z = \begin{bmatrix} z_{(1)} \\ z_{(2)} \\ \vdots \\ z_{(100)} \end{bmatrix} \in \mathbb{R}^{100}$$

- Main advantage of compression: Speeding up learning algorithm and less memory consuming.

Visualization motivation

- Data visualization is an exploratory tool to get insights about the data, and therefore better understand them.
- The problem is that data with more than 3 features cannot be visualized. How to visualize 10,000 dimensions?

  $\Rightarrow$ Dimensionality reduction

Country	GDP (gross domestic product in trillions of US$) $x_{(1)}$	Per capita GDP $x_{(2)}$	Human development index $x_{(3)}$	Life expectancy $x_{(4)}$	Poverty index (Gini) $x_{(5)}$	Mean household income (thousands of US$) $x_{(6)}$
US	14.3	46.7	0.91	78.3	40.8	84.3
Canada	1.5	39.1	0.90	80.7	32.6	67.2
China	5.8	7.5	0.68	73	46.9	10.22
India	1.6	3.4	0.5	64.7	36.8	0.73
Singapore	0.2	56.6	0.8	80	42.5	67.1
Russia	1.4	19.8	0.7	65.5	39.9	0.72

Each country has 6 features

Visualization motivation

- Dimensionality reduction interpretation:

Country	GDP (gross domestic product in trillions of US$) $x_{(1)}$	Per capita GDP $x_{(2)}$	Human development index $x_{(3)}$	Life expectancy $x_{(4)}$	Poverty index (Gini) $x_{(5)}$	Mean household income (thousands of US$) $x_{(6)}$
US	14.3	46.7	0.91	78.3	40.8	84.3
Canada	1.5	39.1	0.90	80.7	32.6	67.2
China	5.8	7.5	0.68	73	46.9	10.22
India	1.6	3.4	0.5	64.7	36.8	0.73
Singapore	0.2	56.6	0.8	80	42.5	67.1
Russia	1.4	19.8	0.7	65.5	39.9	0.72

Dim reduction/ Compression

$\Rightarrow$

Country	Learned feature 1 $z_{(1)}$	Learned feature 2 $z_{(2)}$
US	-4.5	46.7
Canada	1.6	6.7
China	-5.8	7.5
India	1.6	-3.4
Singapore	0.2	3.2
Russia	2.4	-7.0

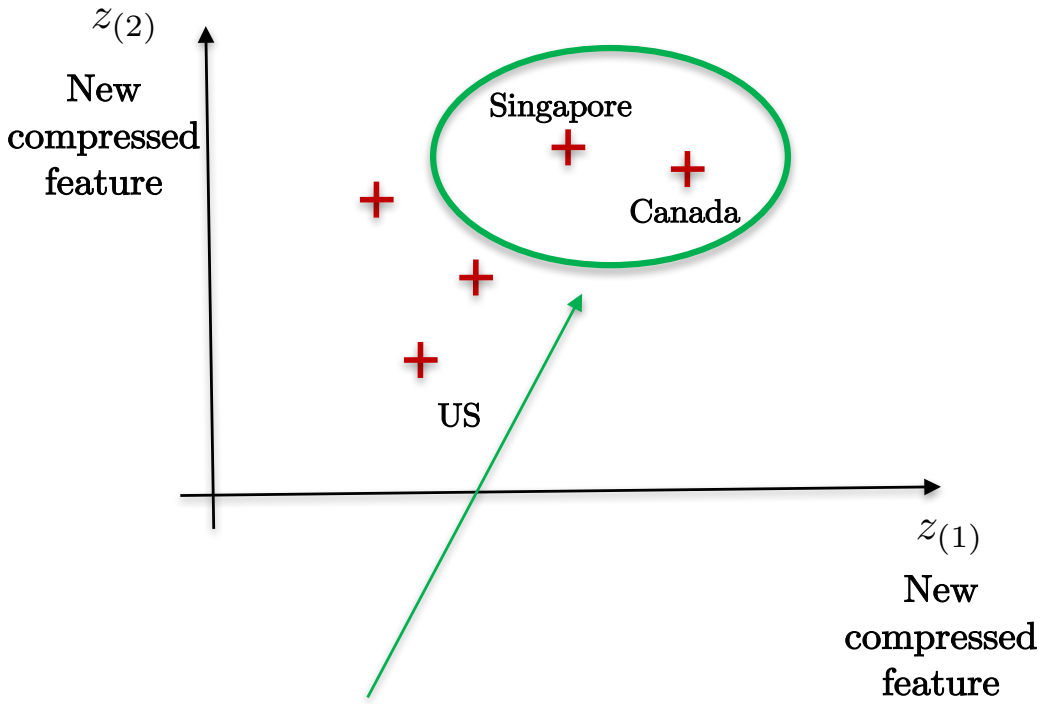
The meaning of hand-crafted features are easy to interpret.

The meaning of learned features can be hard to interpret.

Visualization motivation

- Plot/visualization:

Country	Learned feature 1 $z_{(1)}$	Learned feature 2 $z_{(2)}$
US	-4.5	46.7
Canada	1.6	6.7
China	-5.8	7.5
India	1.6	-3.4
Singapore	0.2	3.2
Russia	2.4	-7.0



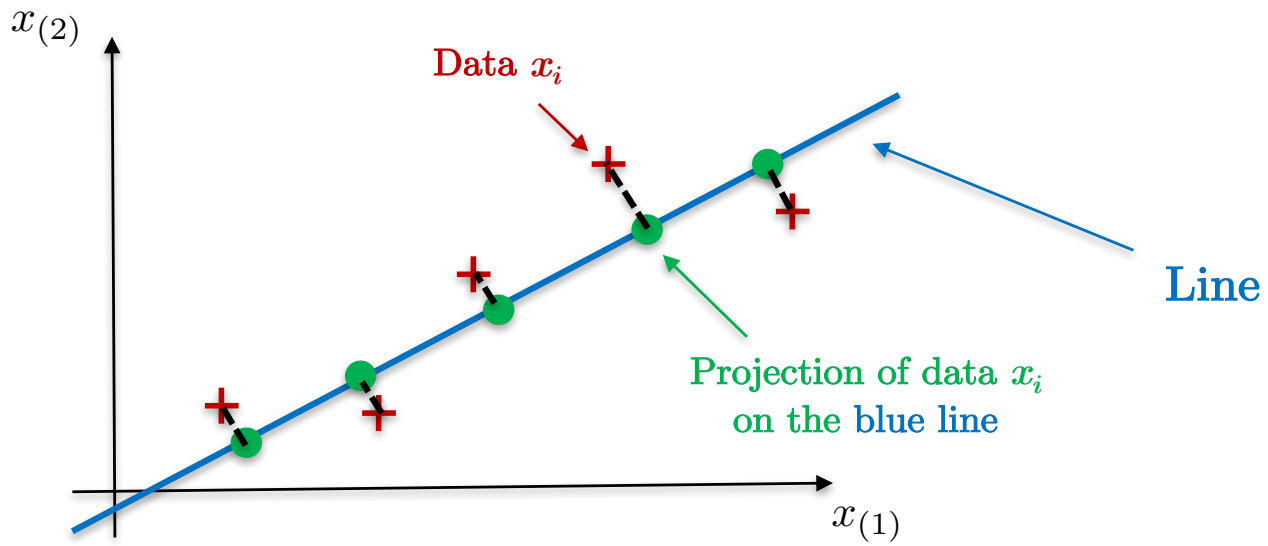
Although interpreting the new features may be hard, the dim reduction technique always put together data that are similar $\Rightarrow$ Clustering property of dim reduction. It makes easier to analyze data.

Outline

- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- **Unsupervised representation with PCA**
  - Introduction and motivations
  - **Principal directions and components**
  - Formalization
  - Algorithm
  - PCA properties
- Conclusion

Basic idea

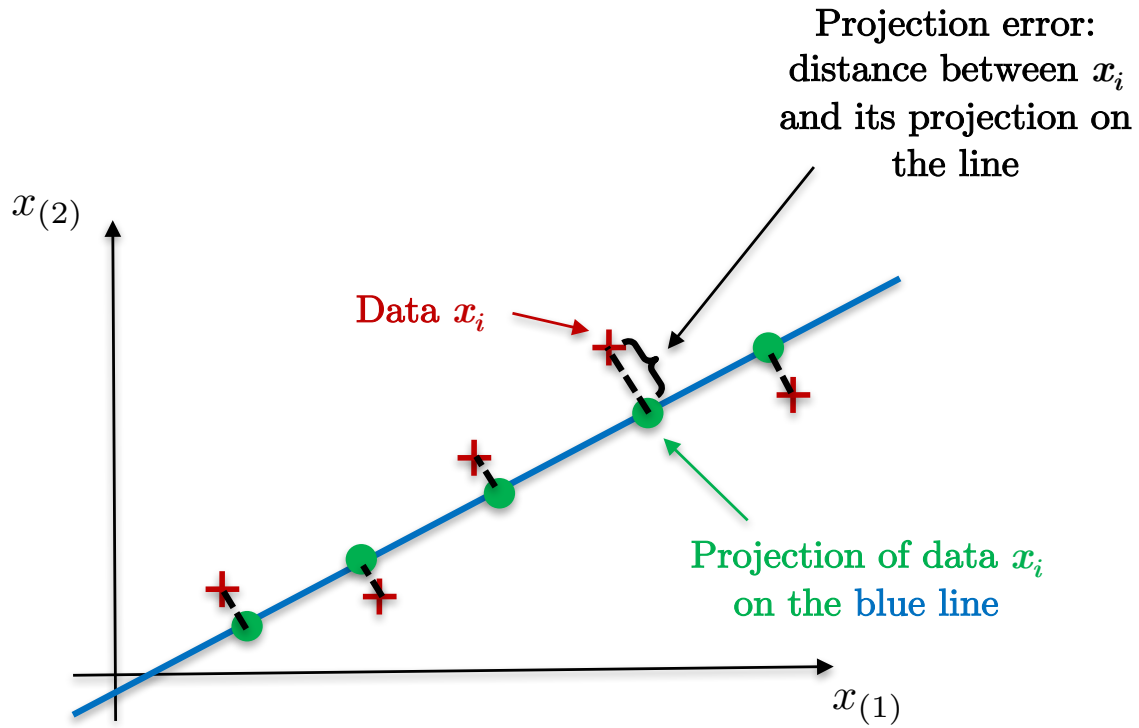
- Reduce the dimensionality of data by projecting the data on the closest line for 1D reduction, the closest plane for 2D reduction, etc.



- How to define a "good" line, a "good" plane, etc?

Basic idea

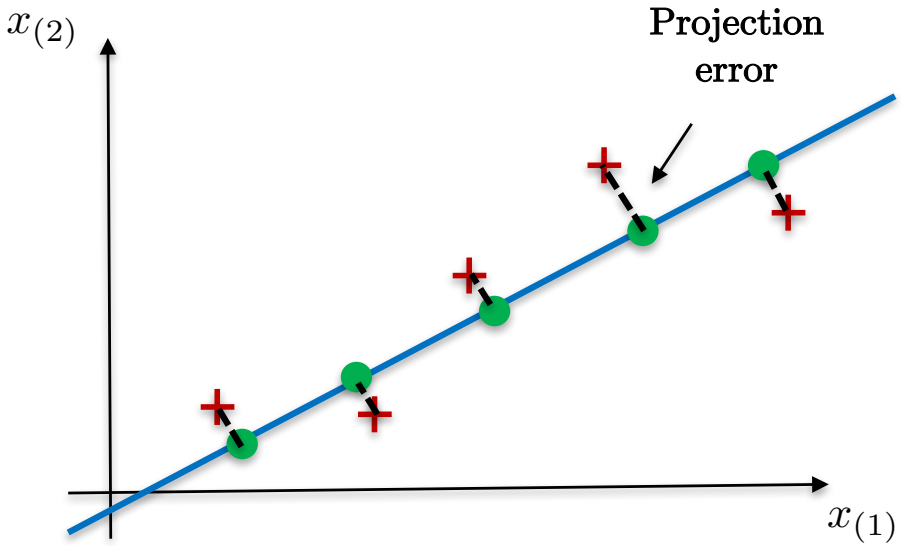
- What is a good line? The line that minimizes as much as possible the projection errors.



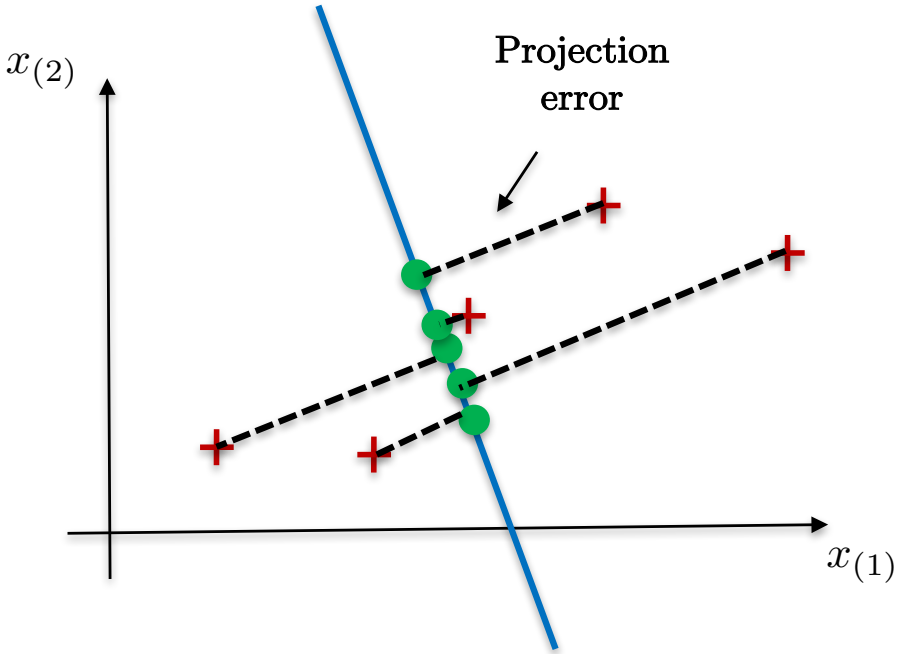
The best blue line if the line that minimizes the sum of all projection errors.

Basic idea

- Different lines:



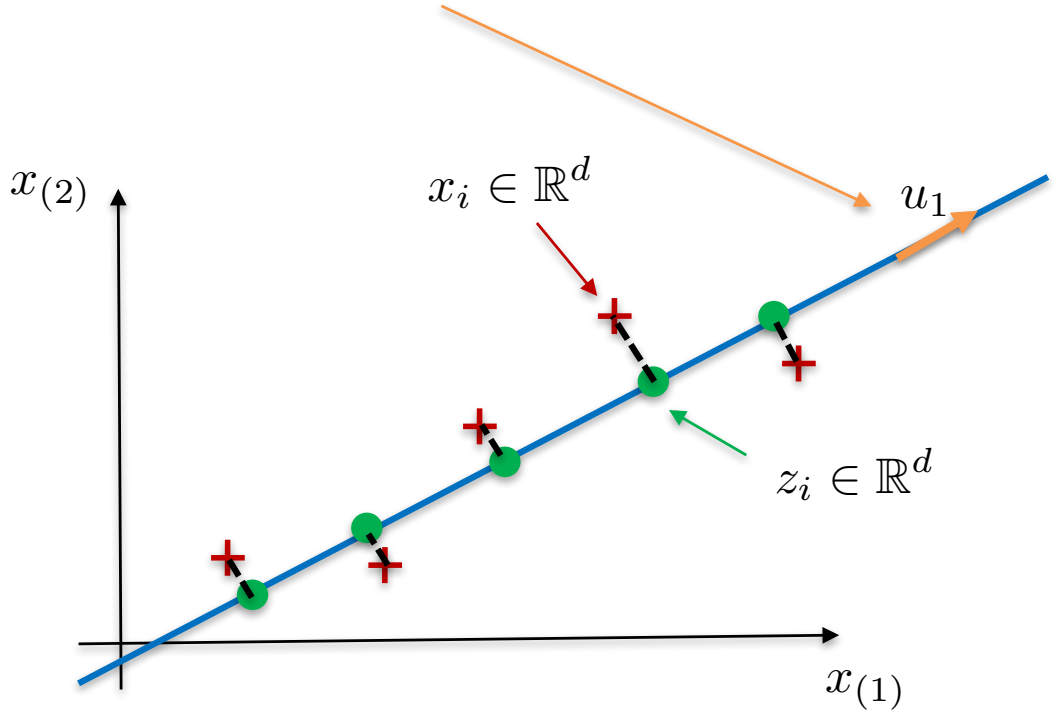
Sum of projection errors is small (and minimized).



Sum of projection errors is high.

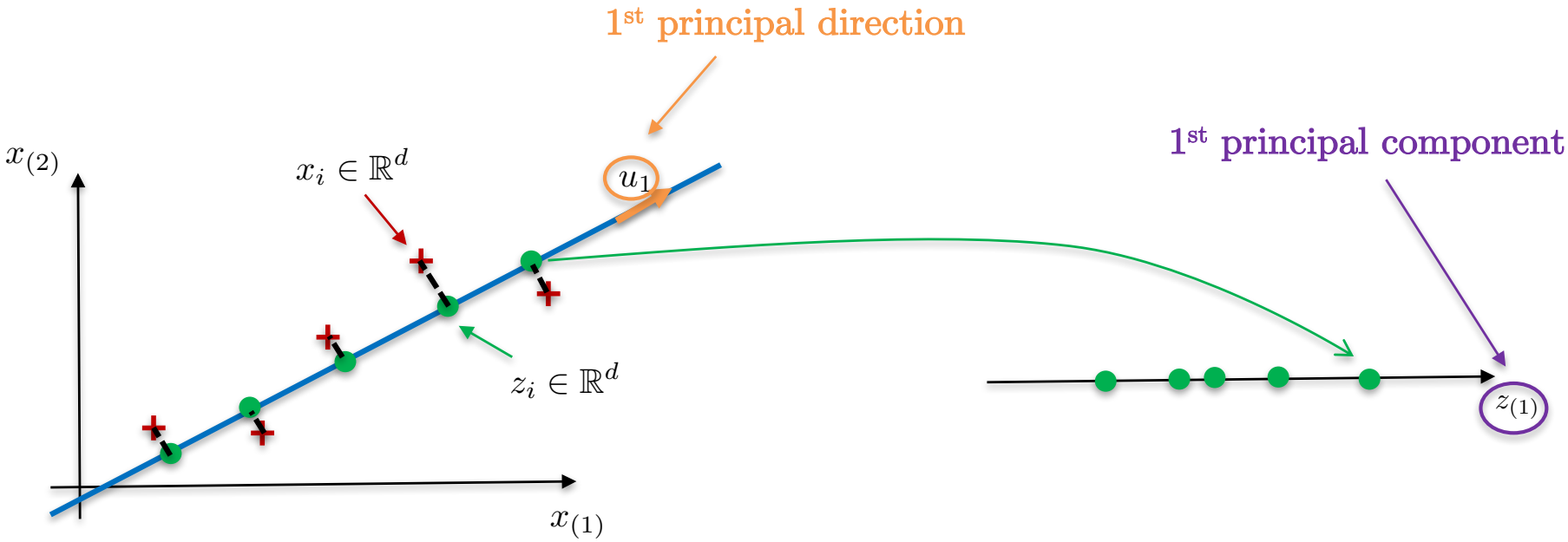
Notations

- x_i : i^{th} data point
- $x_{i(j)}$: j^{th} data feature of the i^{th} data point
- z_i : projection of the i^{th} data point
- $z_{i(j)}$: j^{th} learned feature of the i^{th} projected data point
- u_k : unit vector of the k^{th} principal direction (the line if k=1)



Principal directions and components

- 1st principal direction: The direction u_1 in R^2 onto which the projection of data is minimized and the coordinate $z_{(1)}$ in R^1 on the projected direction u_1.



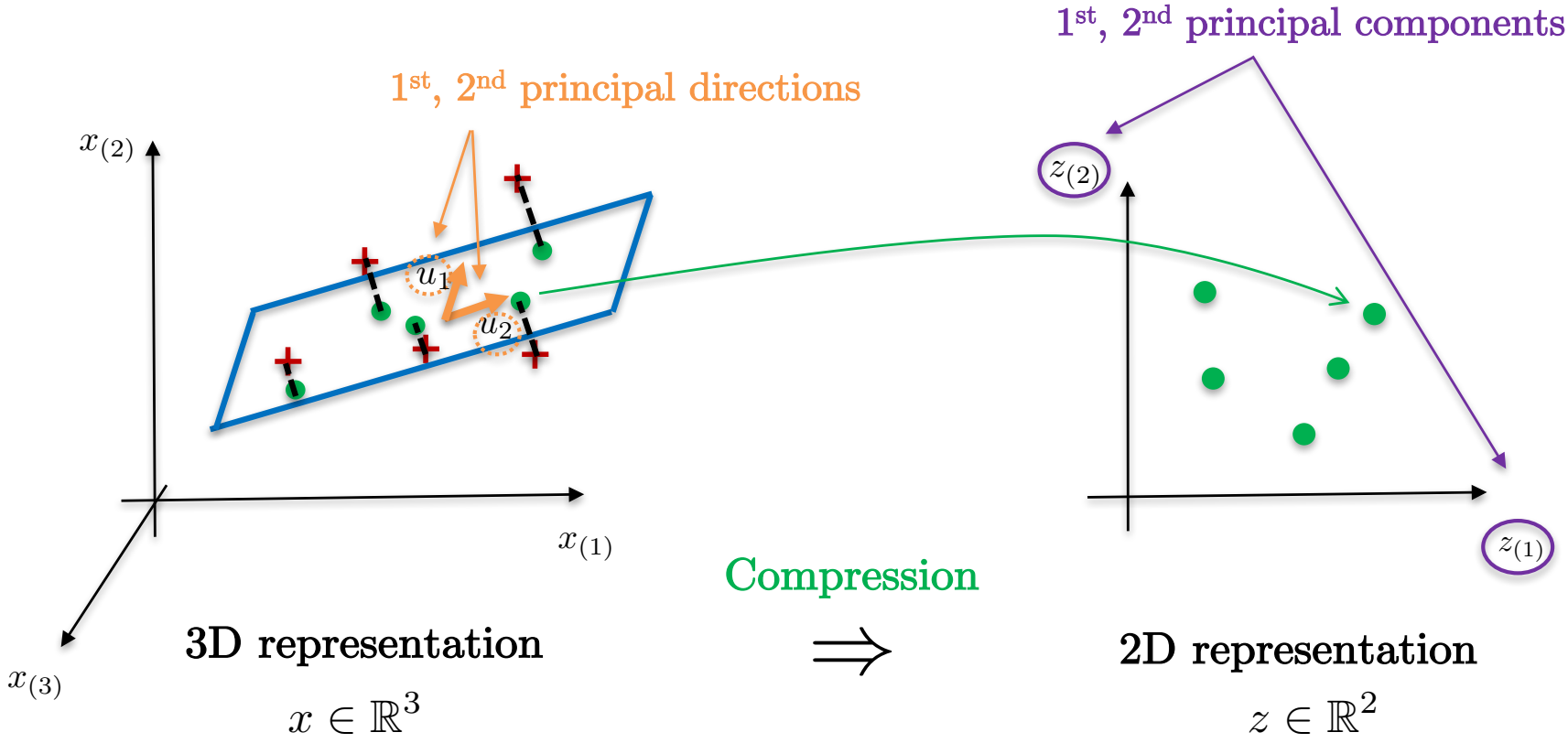
Compression

2D representation $\Rightarrow$ 1D representation

$x \in \mathbb{R}^2$ $\quad$ $z \in \mathbb{R}^1$

Principal directions and components

- 1st, 2nd principal directions: The directions u_1, u_2 in R^3 onto which the projection of data is minimized and the coordinate $z_{(1)}, z_{(2)}$ in R^2 on the projected directions u_1, u_2.



- 1st,...,k^{th} principal directions: The directions $u_1, \ldots, u_k$ in R^d onto which the projection of data is minimized and the coordinate $z_{(1)}, \ldots, z_{(k)}$ in R^k on the projected directions $u_1, \ldots, u_k$.

PCA pre-processing

- Required pre-processing: Data centering (zero mean)
  - Training set: $x_1,\ldots,x_n$
  - Mean vector: $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \in \mathbb{R}^d$
  - Centering data: $x_i \leftarrow x_i - \mu$
- Other pre-processing: z-scoring may help or not.

$$x_i \leftarrow \frac{x_i - \mu}{\sigma}$$

Outline

- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- **Unsupervised representation with PCA**
  - Introduction and motivations
  - Principal directions and components
  - **Formalization**
  - Algorithm
  - PCA properties
- Conclusion

Formalization

- PCA technique finds a linear space (line, plane, hyper-plane) that minimizes the sum of all projection errors.

- How to find this linear space?

  - Linear algebra !

Co-variance matrix

- Matrix of all variances and co-variances:

$$\underset{dxd}{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} \underbrace{\underbrace{x_i}_{dx1}\underbrace{x_i^T}_{1xd}}_{dxd} = \frac{1}{n} X^T X \qquad \text{with data matrix} \quad \underset{nxd}{X} = \begin{bmatrix} —x_1^T— \\ \vdots \\ —x_n^T— \end{bmatrix}$$

Co-variance matrix (encode all data variations)

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \dots & \Sigma_{1d} \\ & \ddots & & \\ \Sigma_{d1} & \Sigma_{d2} & \dots & \Sigma_{dd} \end{bmatrix}$$

$$\begin{cases} \Sigma_{jj} = \sum_{i=1}^{n} x_{i(j)}^2 & \text{data variance along } j\text{-dim} \\ \Sigma_{jl} = \sum_{i=1}^{n} x_{i(j)} x_{i(l)} & \text{data co-variance along } j\text{-dim and } l\text{-dim} \end{cases}$$

Principal directions

- The principal directions $u_1,\ldots,u_k$ in R^d onto which the projection of data is minimized are given by the eigenvalue decomposition (EVD) of the co-variance matrix (no proof given):

$$\underset{dxd}{\Sigma} = \underset{dxd}{U}\underset{dxd}{S}\underset{dxd}{U^T}$$

Eigenvalue decomposition is a matrix factorization technique.
Eigenvector matrix: U
Eigenvalue matrix: S

$$\underset{dxd}{U} = \begin{bmatrix} | & & | & & | \\ u_1 & \ldots & u_k & \ldots & u_d \\ | & & | & & | \end{bmatrix}$$

$$\underset{dxk}{U_k}$$

Matrix of k principal directions $u_1,\ldots,u_k$ in R^d

with $$u_p^T u_q = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$$

$$u_p^T u_p = \|u_p\|_2^2 = 1, \ \forall p \quad \text{(unit vector)}$$

$$u_p^T u_q = 0, \ \forall p \neq q \quad \text{(orthonormal vectors)}$$

Principal directions

- u_1 represents the direction of the largest data variance.
- u_2 represents the direction of the second largest data variance.
- ...
- S_{11} is the value of the data variance along the direction u_1.
- S_{22} is the value of the data variance along the direction u_2.
- ...

$$\underset{dxd}{\Sigma} = USU^T \qquad \text{with} \qquad S = \begin{bmatrix} S_{11} & & & 0 \\ & S_{22} & & \\ & & \ddots & \\ 0 & & & S_{dd} \end{bmatrix}$$

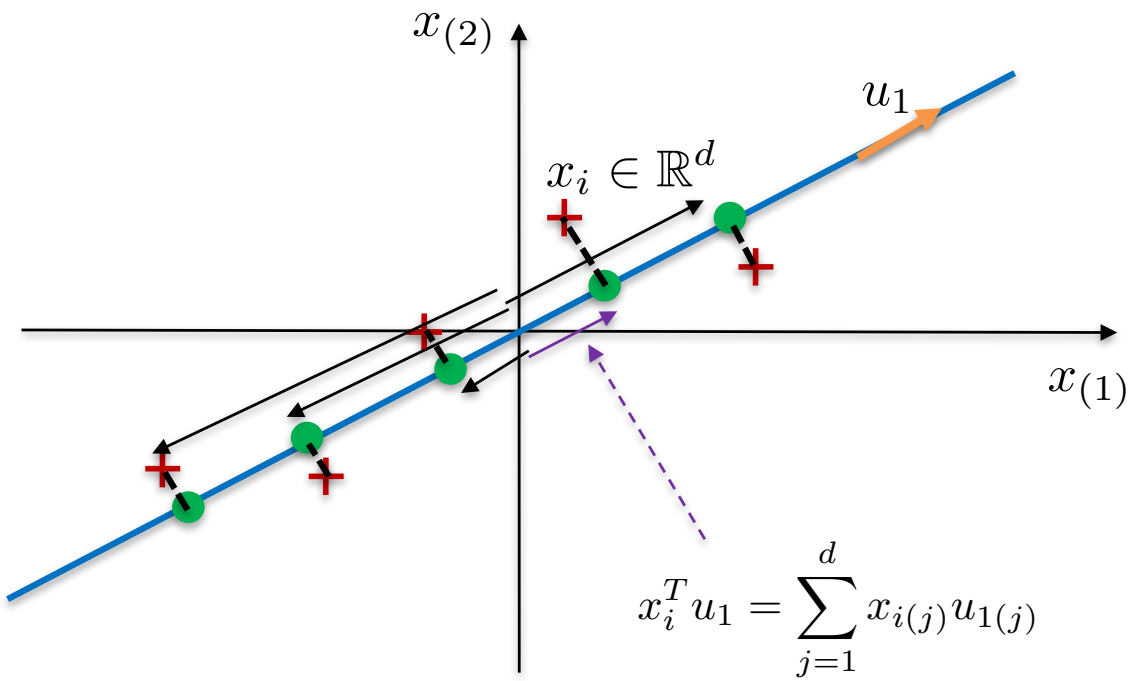
$$\text{and} \qquad S_{ll} = \sum_{i=1}^{n} |x_i^T u_l|^2 \qquad \text{(no proof given)}$$

Principal directions capture the data variance

- S_{11} represents the variance of the data along the direction u_1 of the largest data variance :

$$S_{11} = \sum_{i=1}^{n} |x_i^T u_1|^2$$

Sum of square of projected data along u_1



Outline

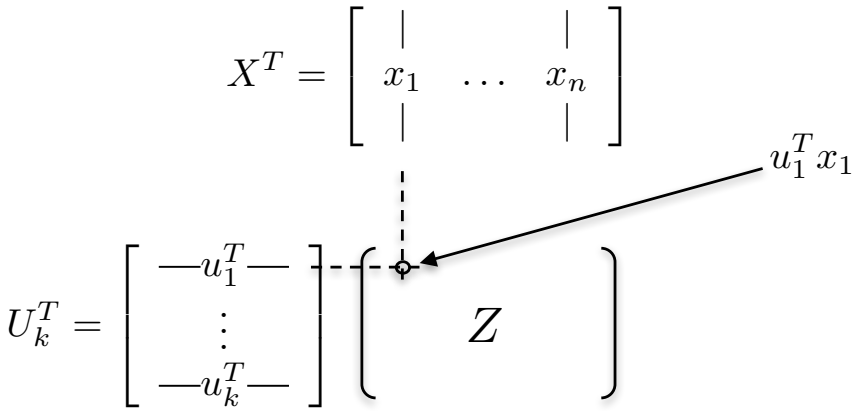
- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- **Unsupervised representation with PCA**
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - **Algorithm**
  - PCA properties
- Conclusion

PCA algorithm

- **Algorithm:**
  - Pre-process data: zero-mean $x_i \leftarrow x_i - \frac{1}{n}\sum_{i=1}^{n} x_i$
  - Construct the co-variance matrix: $\Sigma = \frac{1}{n} X^T X$
  - Compute the EVD of Σ (principal directions): $\Sigma = USU^T$
  - Compute the principal components (or projected data):

$$Z = U_k^T X^T$$
$$k\text{x}n \quad k\text{x}d \quad d\text{x}n$$

Projected data coordinates on $u_1,\ldots,u_k$

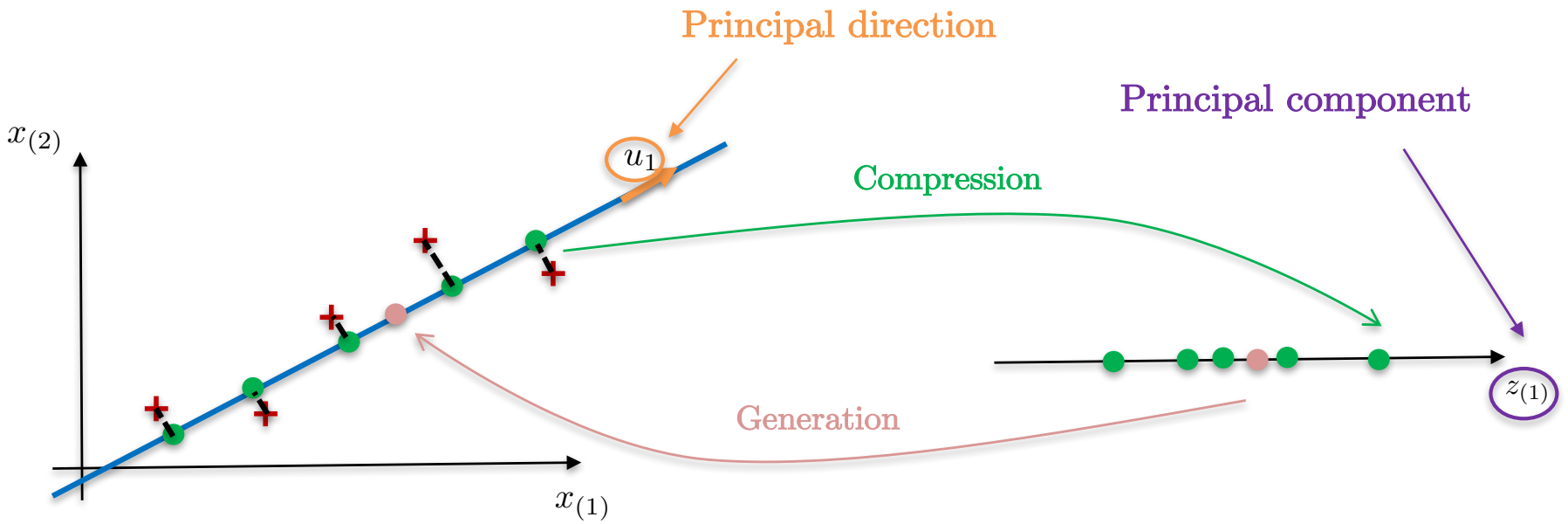


Outline

- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- **Unsupervised representation with PCA**
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - **PCA properties**
- Conclusion

Data generation

- We can construct new data from the compressed representation:



d-D representation
(high-dim representation)
$x \in \mathbb{R}^d$

Compression
$\Rightarrow$

k-D representation
(low-dim representation)
$z \in \mathbb{R}^k$

Generation
$\Leftarrow$

$\hat{x} = U_k \hat{z} \ \in \mathbb{R}^d$

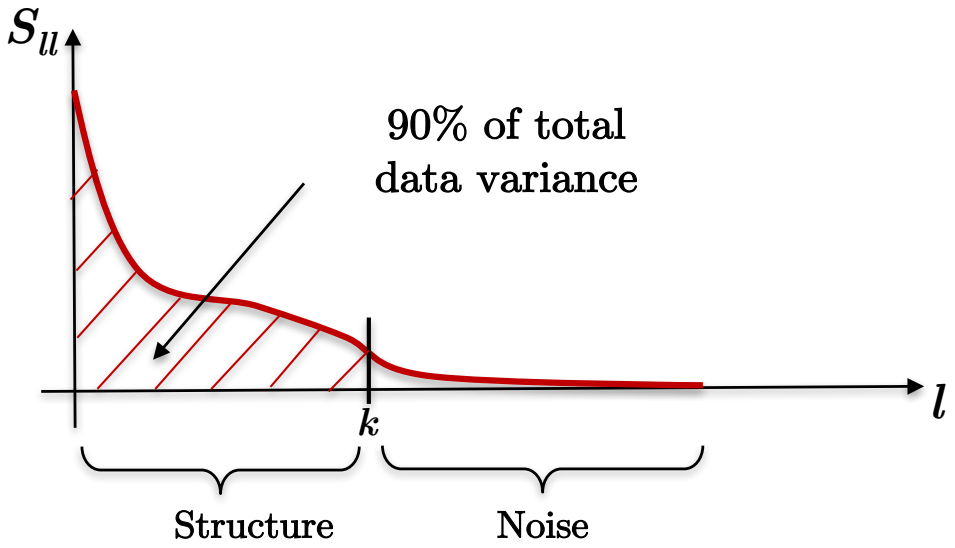
$\hat{z} \in \mathbb{R}^k$

How to select k?

- Selection rule: Data variations are captured by each principal direction $\Rightarrow$ A natural rule is to select the first k principal directions that capture e.g. 90% of total variance:

$$\frac{\sum_{l=1}^{k} S_{ll}}{\sum_{l=1}^{d} S_{ll}} \geq 0.9$$

S_{ll} represents the variance of the data along the direction u_1



PCA in practice

- Applications of PCA:
  - Visualization:
    - Insights about data properties
  - Compression:
    - Reduce memory needed for data
    - Speed up learning speed

$$(x_1, y_1), \ldots, (x_n, y_n), x_i \in \mathbb{R}^d$$

$$\Downarrow$$

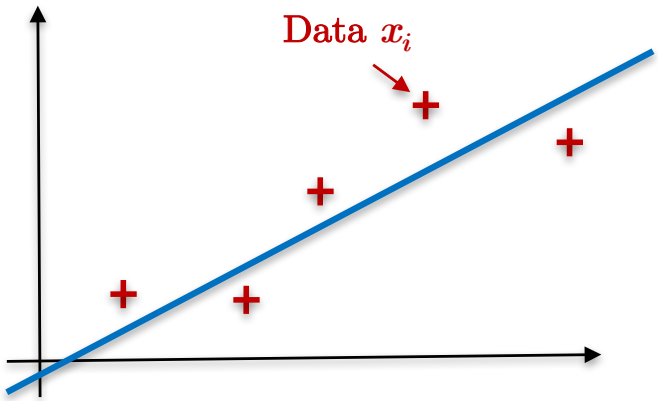
$$(z_1, y_1), \ldots, (z_n, y_n), z_i \in \mathbb{R}^k$$

  - Example: Images d=256x256 $\Rightarrow$ k=128

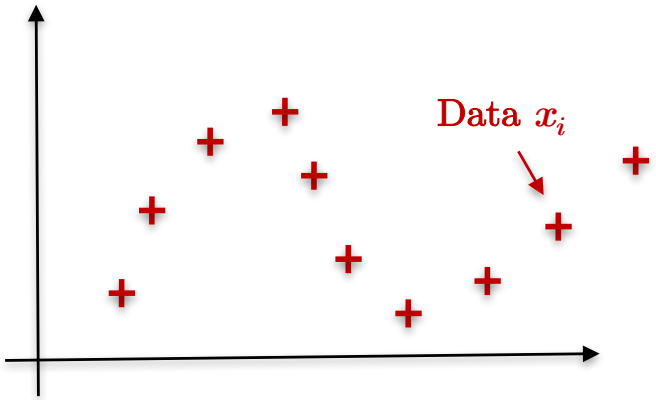
- Important: There is no guarantee that applying PCA will improve the performances of learning algorithm. Actually, PCA usually decreases the results, but it is fine if speed is a priority.

Limitation

- PCA is a linear dimensionality reduction technique $\Rightarrow$ It means that it is guaranteed to work when the data follow a linear distribution, but no guarantee otherwise.



Linear distribution



Non-linear distribution

Summary

- PCA is the most common and the oldest linear dimensionality reduction technique.
- There exit several improvements of PCA: sparse PCA, robust PCA, graph PCA, non-linear PCA, etc.
- PCA algorithm is simple, sound, linear w.r.t. number of training data and number of principal components, but does not scale w.r.t. number of features d (complexity is $O(d^3)$).
- PCA is optimal for data with linear distribution.

Outline

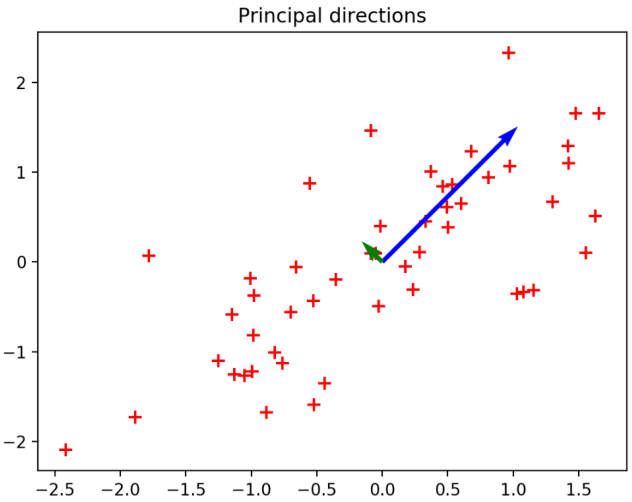
- Supervised vs unsupervised learning
- Unsupervised clustering with k-means
  - Clustering
  - k-means algorithm
  - Loss
  - k-means properties
- Unsupervised representation with PCA
  - Introduction and motivations
  - Principal directions and components
  - Formalization
  - Algorithm
  - PCA properties
- **Conclusion**

Conclusion

- Unsupervised learning techniques solve data analysis tasks without any label information.
- Unsupervised techniques can be applied to data clustering, data representation (beyond linear PCA s.a. generative adversarial networks (GAN)), data visualization(beyond PCA s.a. t-SNE), etc.
- Unsupervised learning problems are much more challenging to solve than supervised learning problems.
- The next AI revolution will be unsupervised [LeCun].

Coding exercise

- tutorial07.ipynb



1.5 Main principal directions

Data variations are captured by each principal direction. A natural rule is to select the first k principal directions that capture e.g. 85% of total variance:

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{d} S_{ii}} \geq 0.85$$

How many principal directions do we need to capture 85% of the total data variance?

Questions?