Apprentissage supervisé (suite) : Réseaux de Neurones Apprentissage non-supervisé : Clustering

Reconnaissance des Formes (Semaine VIII)

Université de Lille, France

13 mars 2013

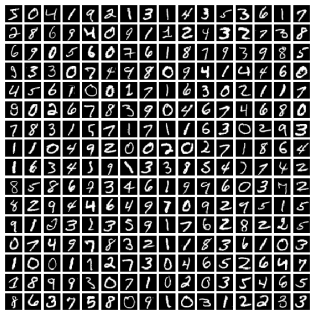
Rappel

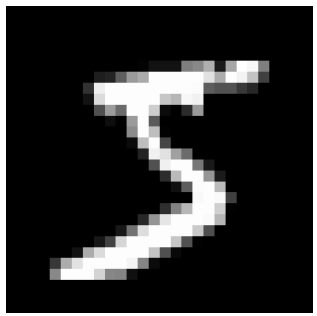
Apprentissage Supervisé

L'objectif est de trouver la structure/forme dans les données, en utilisant quelques exemplaires pour lesquels la solution est connue.

Exemple : Classification

- L'ensemble des données est partagé en quelques classes différentes,
- On a quelques exemplaires pour qui on sait les classes correctes.
- On veut classifier correctement les données pour qui les classes correctes sont inconnus.





Algorithmes d'apprentissage supervisé déjà étudiés en cours

1. k-Nearest Neighbors ou k plus proches voisins
   Approche : Classifier chaque point $\mathbf{x}$ par la vote à la majorité de ses k voisins les plus proches

Algorithmes d'apprentissage supervisé déjà étudiés en cours

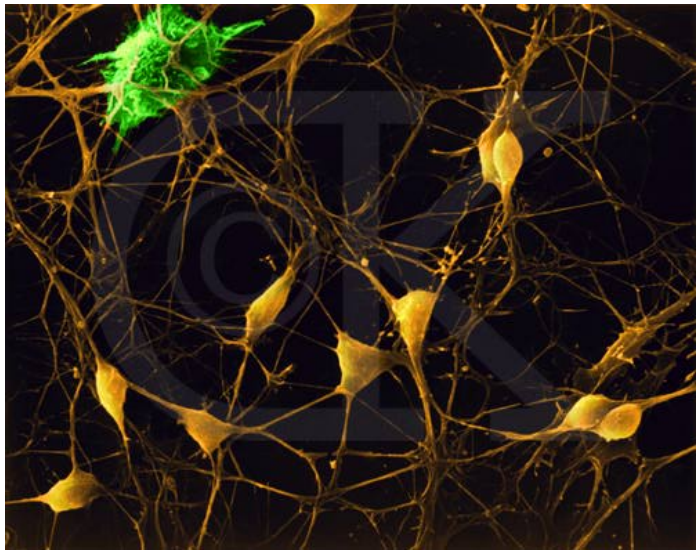
1. k-Nearest Neighbors ou k plus proches voisins
   Approche : Classifier chaque point $\mathbf{x}$ par la vote à la majorité de ses k voisins les plus proches
2. Naïve Bayes : Un modèle probabiliste qui consiste à choisir la classe la plus probable pour chaque point $\mathbf{x}$
   - **Règle de décision :** Maximum a Posteriori (MAP)
   - **l'hypothèse principale :** Indépendance Conditionnelle

Algorithmes d'apprentissage supervisé déjà étudiés en cours

1. k-Nearest Neighbors ou k plus proches voisins
   Approche : Classifier chaque point $\mathbf{x}$ par la vote à la majorité de ses k voisins les plus proches
2. Naïve Bayes : Un modèle probabiliste qui consiste à choisir la classe la plus probable pour chaque point $\mathbf{x}$
   - **Règle de décision :** Maximum a Posteriori (MAP)
   - **l'hypothèse principale :** Indépendance Conditionnelle
3. Régression Logistique (pour classification en deux classes 0 ou 1)
   Approche : Pour chaque $\mathbf{x}$ calculer $h_\theta(\mathbf{x}) := \frac{1}{1+e^{-\boldsymbol{\theta}^T \mathbf{x}}}$
   - **Prédiction :** Classifier $\mathbf{x}$ dans la classe 1 si $h_\theta(\mathbf{x}) \geq 0.5$ et dans la classe 0 si $h_\theta(\mathbf{x}) < 0.5$
   - **Trouver $\boldsymbol{\theta}$:** Utiliser les exemplaires pour trouver le $\boldsymbol{\theta}$ qui minimise une fonction de coût $J(\theta)$

Réseaux de Neurones : Un autre Modèle pour apprentissage supervisé

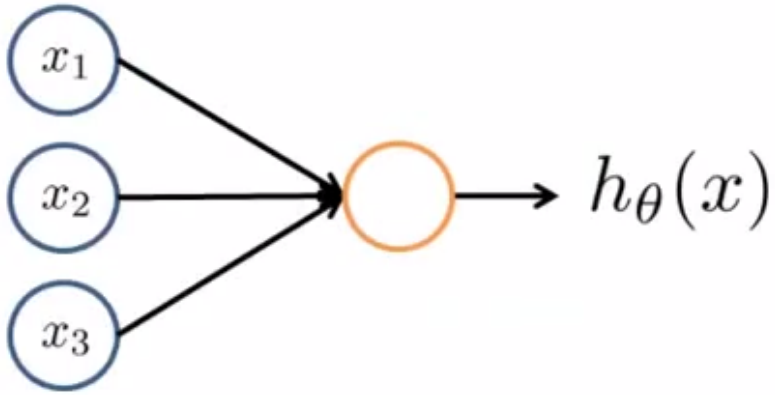
- Algorithmes qui essayent d'imiter le cerveaux en apprentissage



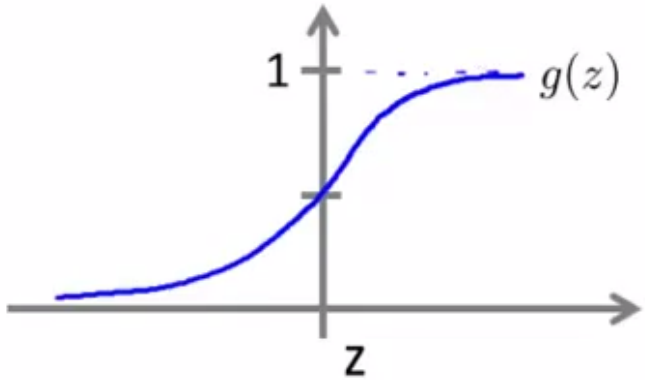
- Utilises dans les années 80 et 90, mais bondonnés vers la fin des années 90 !
  - $\rightarrow$ peut-être car ils demandent beaucoup de capacité de calculs
- Retour en grâce : énormément utilisées pour des nombreuses applications !
- Il nous permet de trouver les frontières de décision non-linaires

Modéliser un Neurone

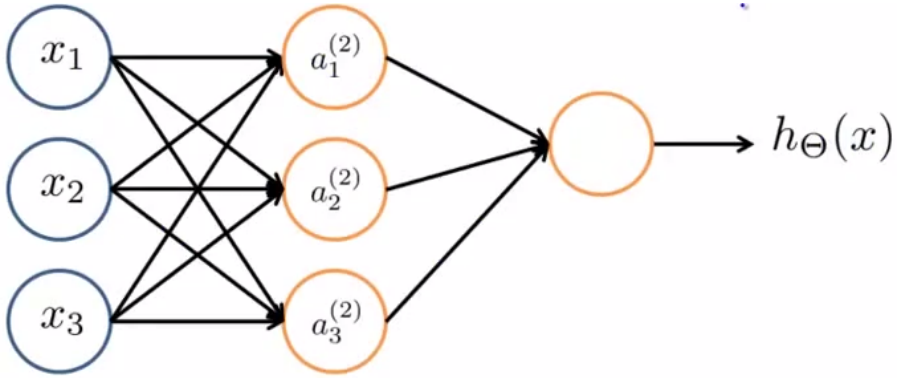
Neurone est modéliser comme un unit logistique



où $h_\theta(\mathbf{x}) := g(\boldsymbol{\theta}^T \mathbf{x})$ et $g(z) := \frac{1}{1+e^{-z}}$ est la fonction d'**activation** sigmoid ou logistique :



Modéliser le Réseaux de Neurones



(les couches intermédiaires s'appellent les couches cachés)

Paramètres

- $a_i^{(j)}$: La fonction d'activation i à couche j
- $\mathbf{\Theta}$: Une **matrice** des poids qui contrôle les activations entre couches j et $j+1$

Apprentissage (par l'algorithme BackPropagation)

De manière similaire à régression logistique l'on utilise les données d'entraînement pour minimiser une fonction de coût $J(\mathbf{\Theta})$

Réseaux de Neurones en Action

Demo : par Yann Lecun

Apprentissage Non-Supervisé

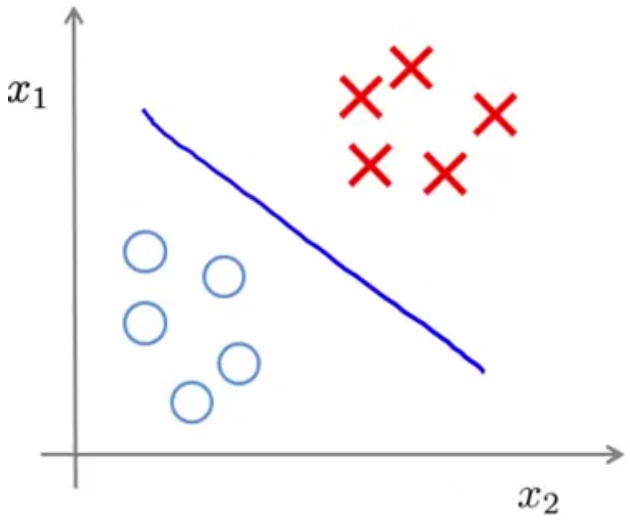
Objectif

Trouver une structure/forme dans les données sans exemplaires ou l'ensemble d'entraînement

Exemple : Clustering → Classification sans données d'entraînement

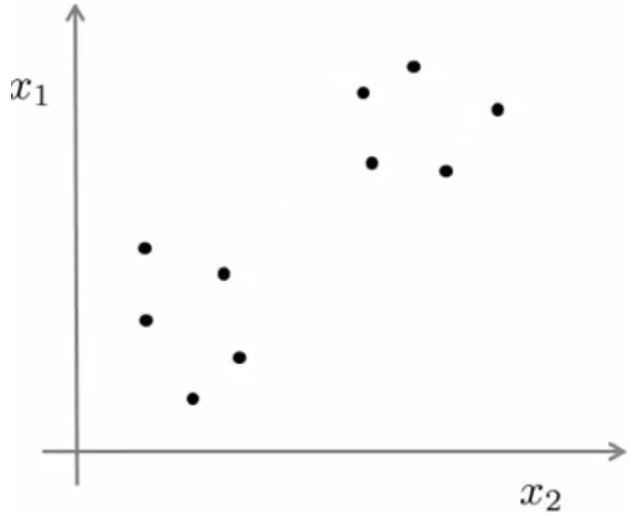
Classification

entrée :
$\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})\}$

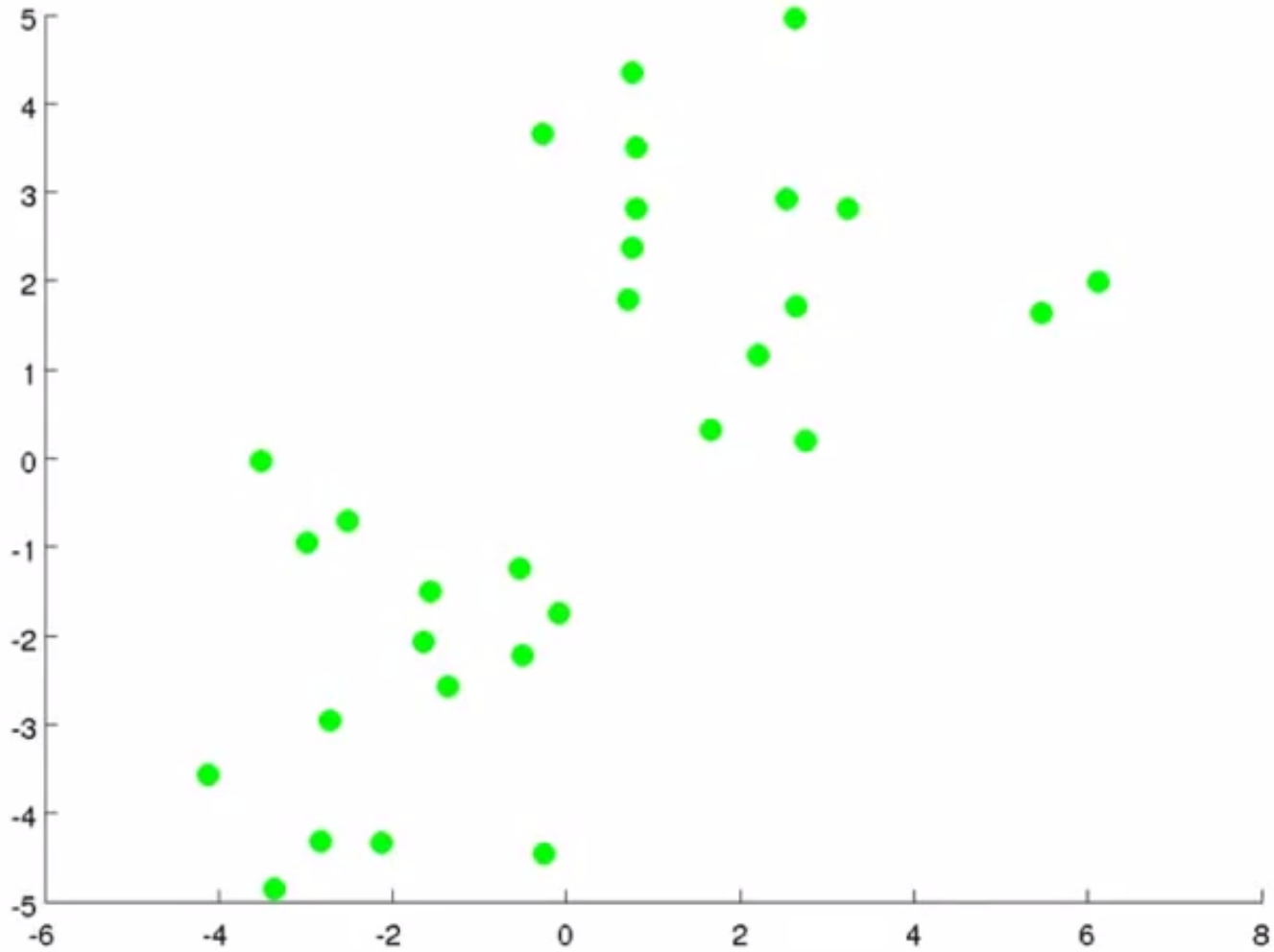


Clustering

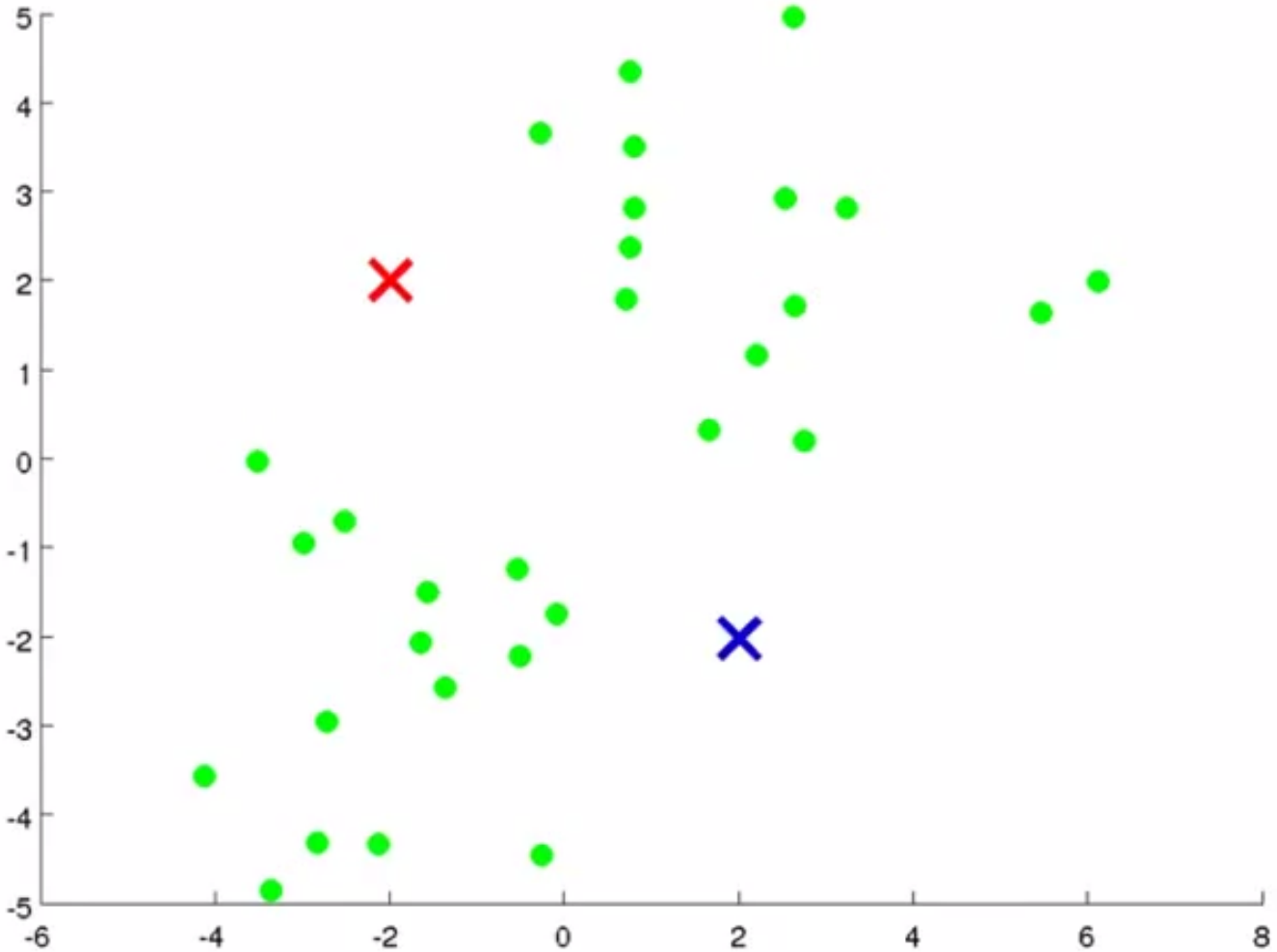
entrée : $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$



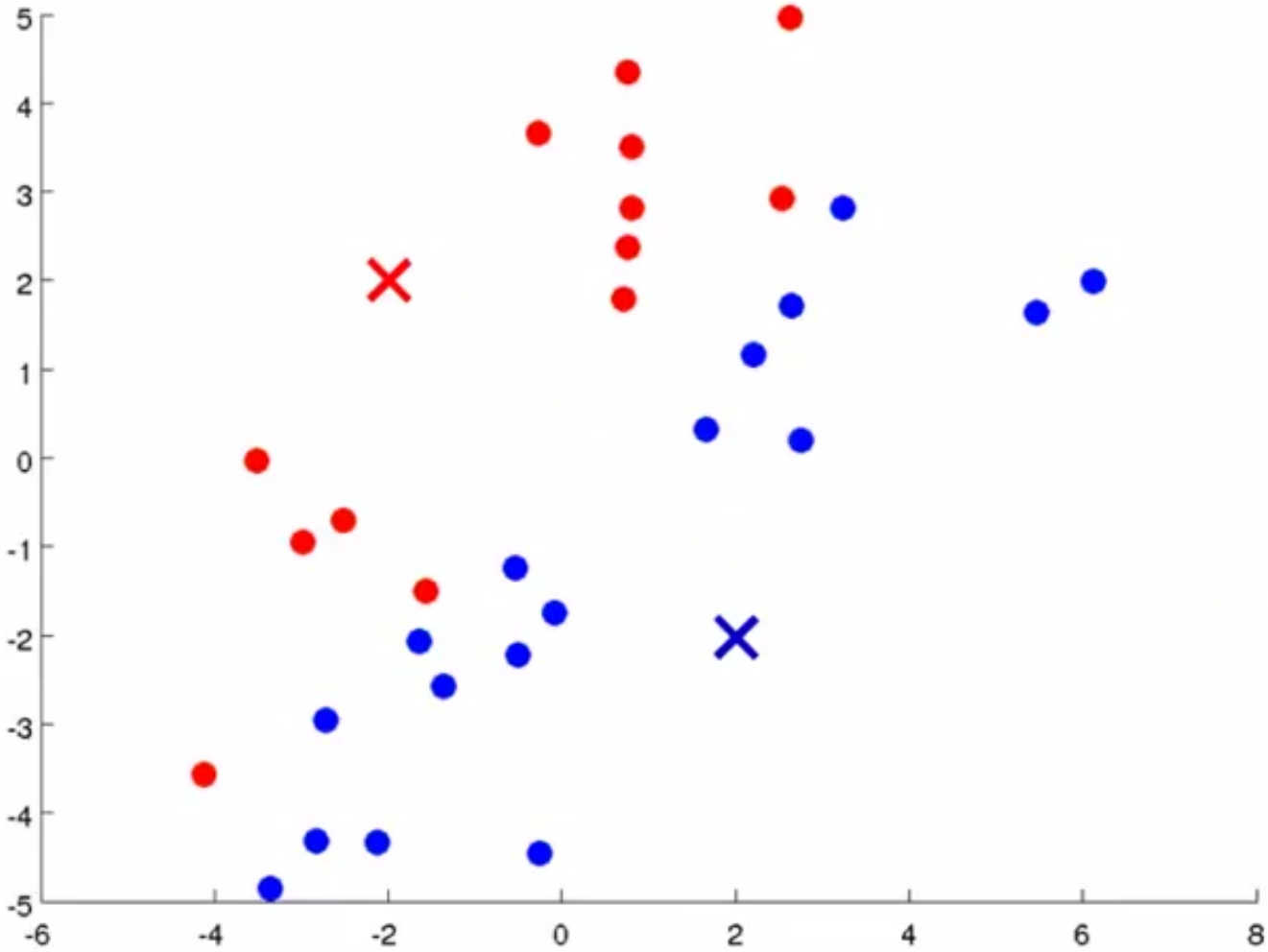
L'algorithme k-Means



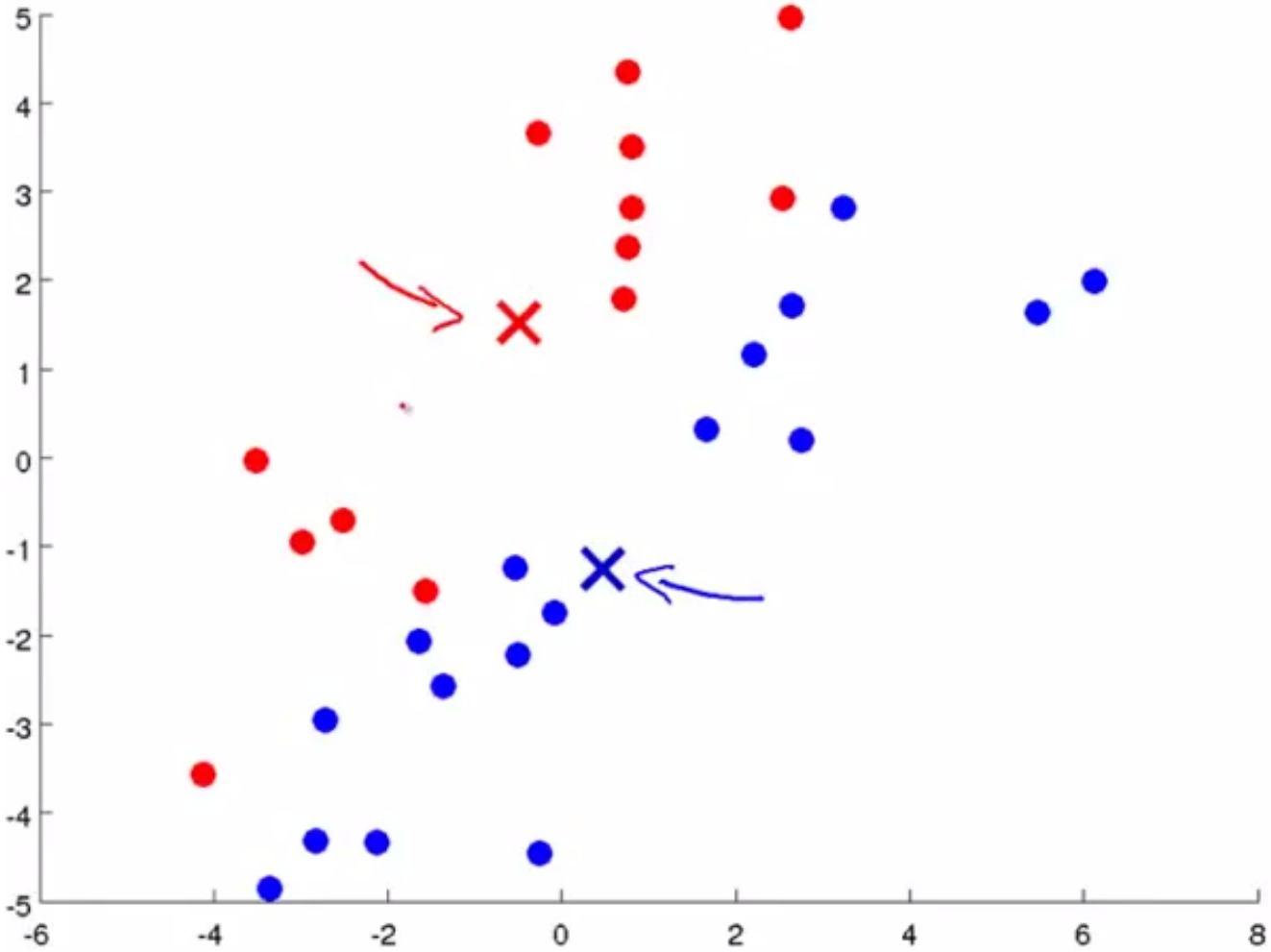
L'algorithme k-Means



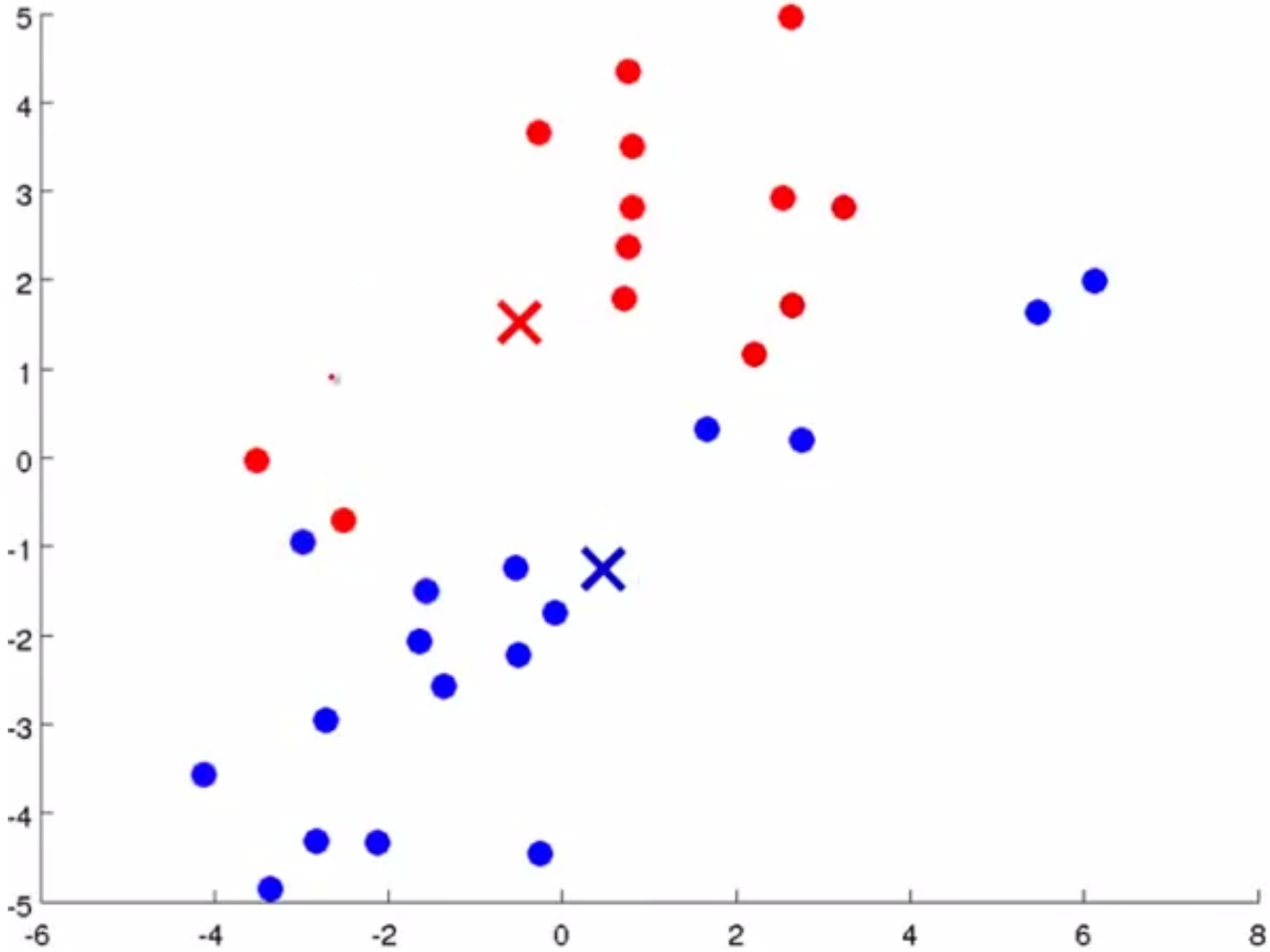
L'algorithme k-Means



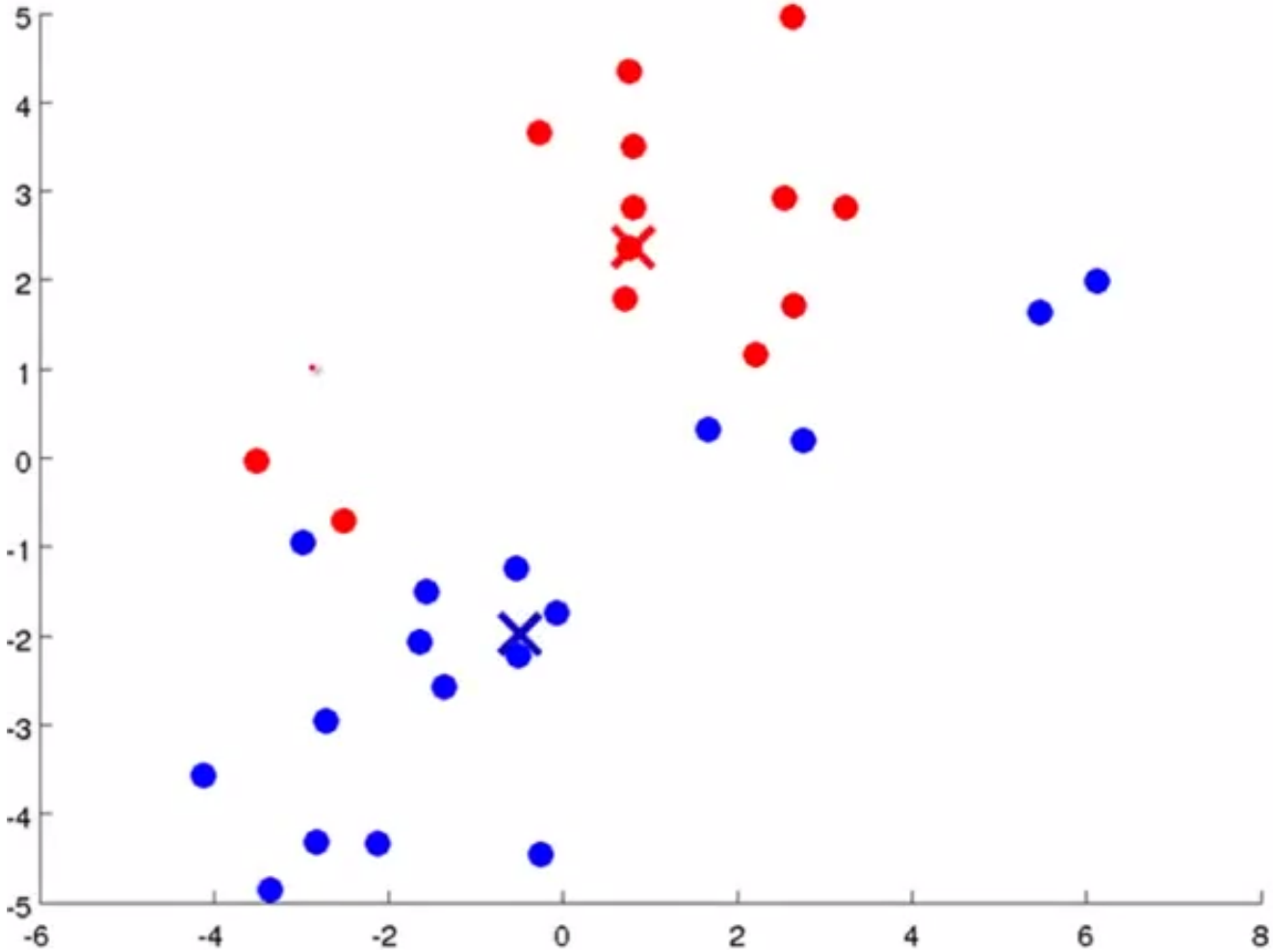
L'algorithme k-Means



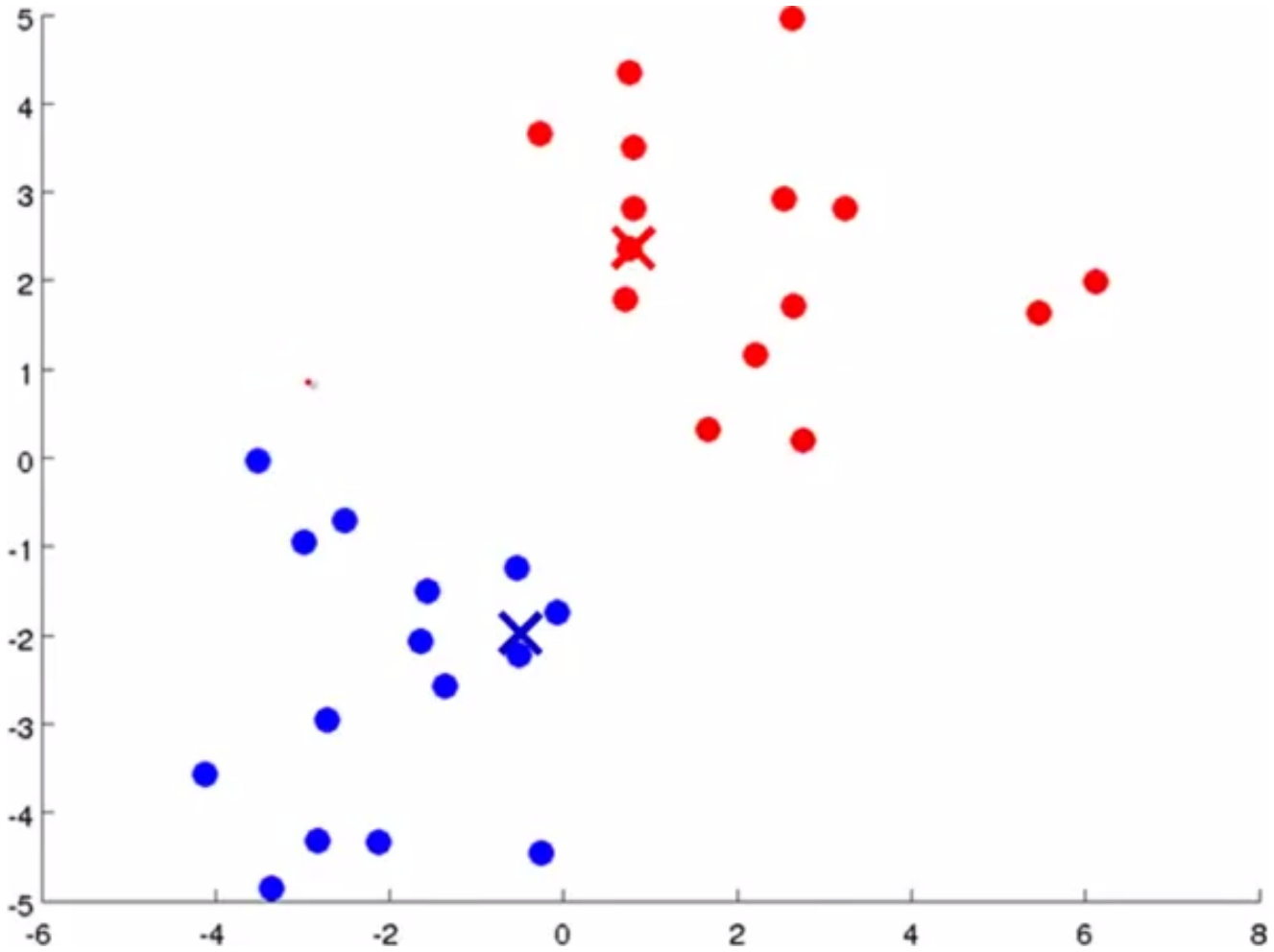
L'algorithme k-Means



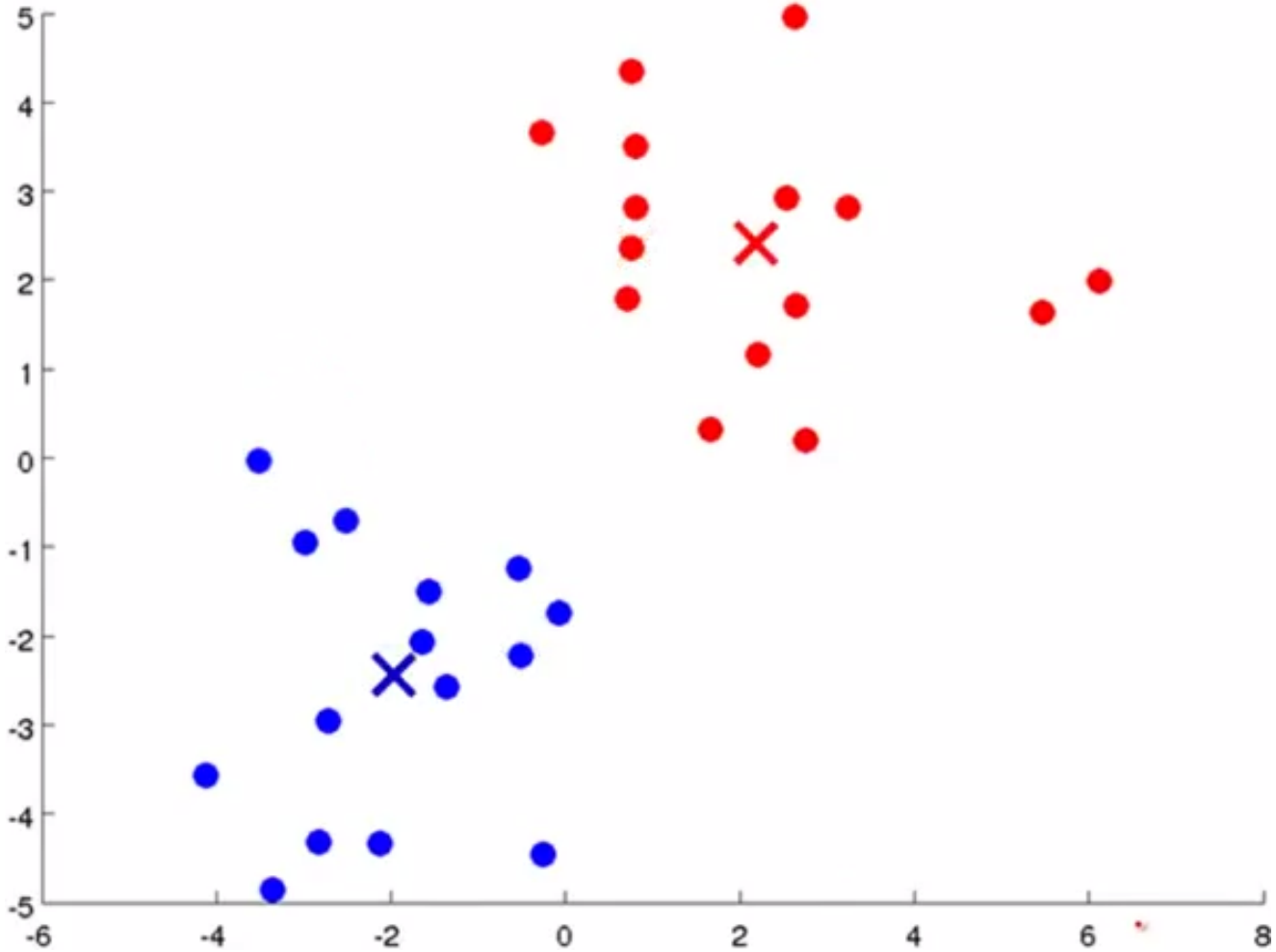
L'algorithme k-Means



L'algorithme k-Means



L'algorithme k-Means



L'algorithme k-Means

Entrées :
K := nombres des clusters (groups)
Données à grouper : $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}\}$

1. Initialiser aléatoirement K centre du cluster $\mu_1, \mu_2, \ldots, \mu_K$
2. Répéter
   1. **[i.] Attribuer chaque point $\mathbf{x}^{(i)}$ à un des K clusters :**
      for $i = 1..m$

      $$c^{(i)} \leftarrow \text{l'indice (entre } 1..K\text{) du centre le plus proche de } \mathbf{x}^{(i)}$$

      endfor
   2. **[ii.] Mettre à jour les centres :**
      for $k = 1..K$

      $$\mu_k \leftarrow \text{la moyenne des points attribués à cluster } k$$

      endfor

L'algorithme k-Means

Remarque

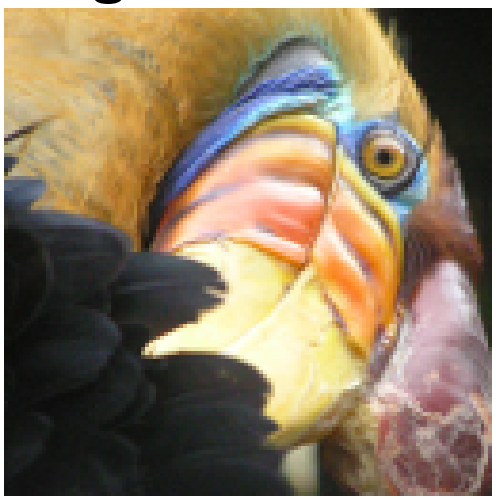
- Normalement l'initialisation des centres est fait aléatoirement
- L'algorithme est arrêté à partir du moment où les centres ne changent plus

Application pour Traitement d'image

Compression d'image par K-means

- Dans le format de codage des couleurs RGB (Red, Green, Blue), chaque pixel est représenté par trois entiers entre 0..255, qui spécifient l'intensité des couleurs Rouge, Vert et Bleu.
- Normalement, chaque image est composé de nombreuse couleurs
  - $\rightarrow$ Une façon de représenter un image de manier efficace est de réduire le nombre des couleurs dans l'image

Original



Compressé (16 couleurs)

Compression d'image par K-means

Approche

- Traiter chaque pixel comme un vecteur de donnée de dimension trois, i.e. chaque pixel est $\mathbf{x} \in \mathbb{R}^3$
- Utiliser K-means pour grouper les pixels en K clusters de couleurs
- Remplacer chaque pixel dans l'image original par le centre de son cluster

Remarque : On utilise la distance Euclidienne.
Demo !