Apprentissage automatique
Machine learning

Fabrice Lembrez - PSI*

Lycée Pierre de Fermat

Objectif poursuivi

Le programme d'informatique et celui de SII font mention de l'apprentissage automatique (machine learning) qui est une branche de l'intelligence artificielle. Mais aucune étude systématique n'est prévue.

Mes objectifs

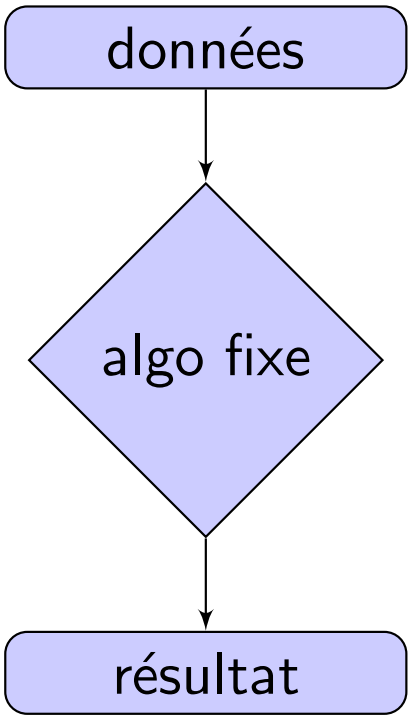
- définir et confronter apprentissage supervisé ou non supervisé
- étudier les exemples proposés par les programmes.
- évoquer l'évaluation de performance, très technique, seulement sur ces exemples.

Plan

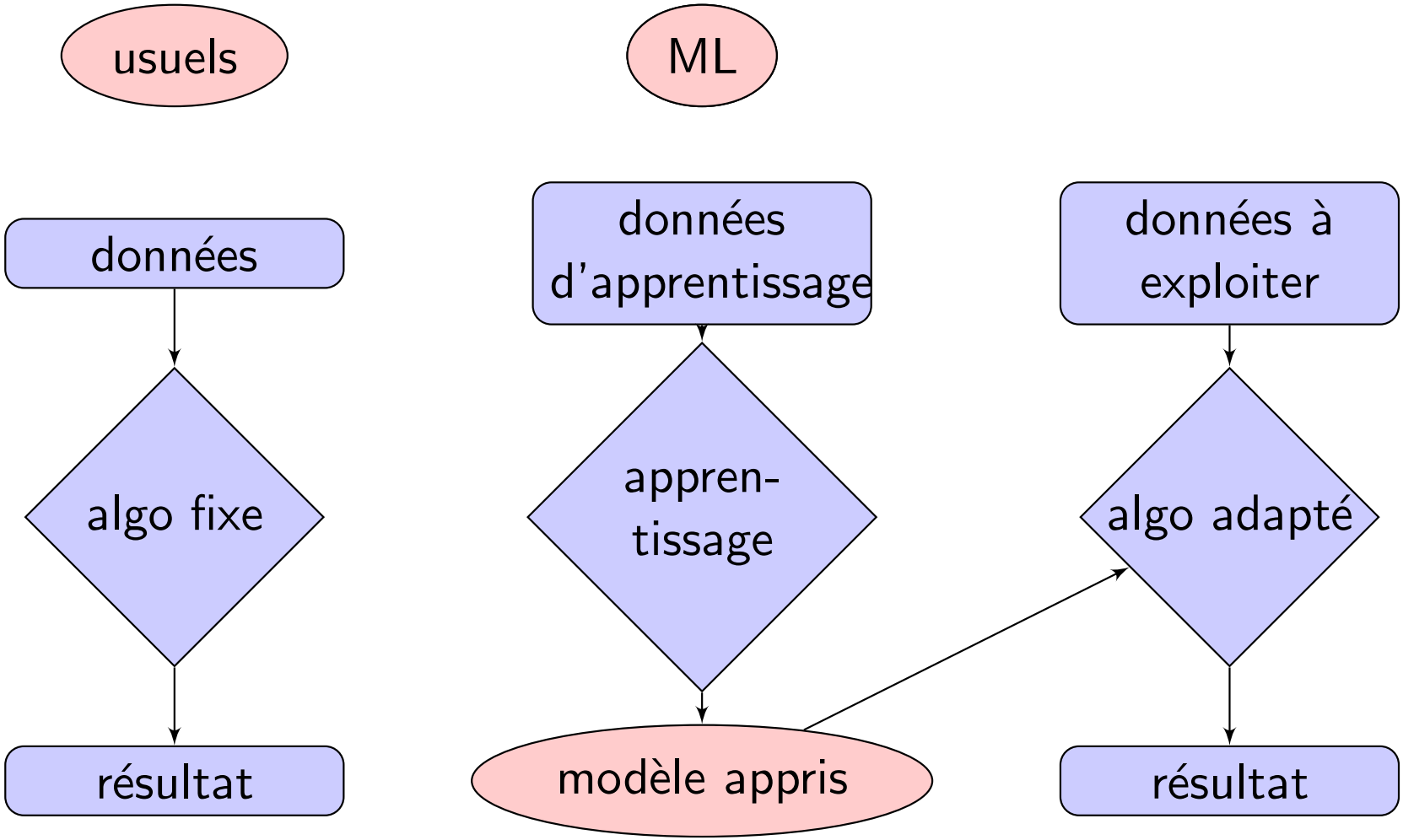
Algorithmes

usuels

ML



Algorithmes



Utilité

C'est pertinent quand on a
- un nombre important de données,
- pas de traitement connu pour les exploiter, ou pas de traitement raisonnable

Avec le modèle on pourra
- analyser les traits pertinents des données existantes,
- produire des prédictions.

Exemples : analyse d'image, modélisation du comportement d'usagers ou de clients, classification d'espèces végétales, aide au diagnostic...

Tentative de définition

Déf - Apprentissage automatique (machine learning)

L'apprentissage automatique consiste à utiliser les données comme source d'amélioration de la performance à résoudre certaines tâches.

On peut y voir une sous-partie de l'intelligence artificielle (on aide la machine à "apprendre" à partir des données), ou comme un domaine qui se chevauche avec l'IA mais qui était déjà pratiqué avant de parler d'IA, avec d'autres idées directrices (notamment l'idée d'optimisation).

Apprentissage supervisé ou non

Apprentissage supervisé à partir de données étiquetées

- on a un stock de données "exemples" avec une "réponse" connue
- "apprentissage" : construire un modèle,
- mesurer sa performance prédictive sur les exemples,
- puis on peut faire des "prédictions".

Apprentissage supervisé ou non

Apprentissage supervisé à partir de données étiquetées

- on a un stock de données "exemples" avec une "réponse" connue
- "apprentissage" : construire un modèle,
- mesurer sa performance prédictive sur les exemples,
- puis on peut faire des "prédictions".

Apprentissage non supervisé

- on a un "bloc de données", sans information a priori.
- but : réduire l'information à ses traits principaux.
- forger néanmoins des tests de qualité pour une telle réduction.

Apprentissage supervisé ou non

Apprentissage supervisé à partir de données étiquetées

- on a un stock de données "exemples" avec une "réponse" connue
- "apprentissage" : construire un modèle,
- mesurer sa performance prédictive sur les exemples,
- puis on peut faire des "prédictions".

Apprentissage non supervisé

- on a un "bloc de données", sans information a priori.
- but : réduire l'information à ses traits principaux.
- forger néanmoins des tests de qualité pour une telle réduction.

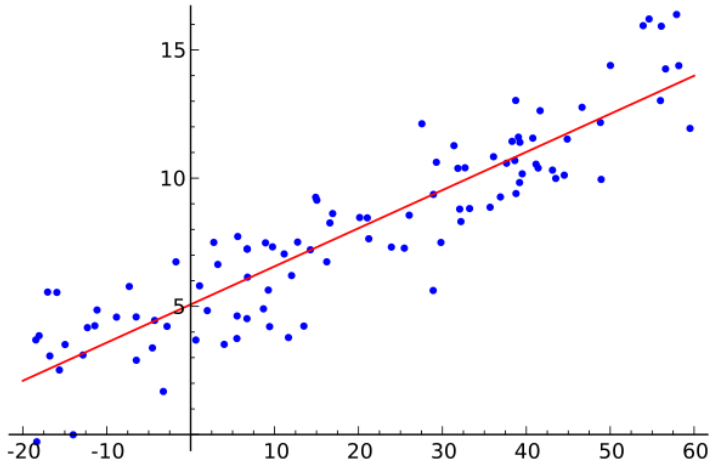
Dans les deux cas il faut tester la qualité du modèle prédictif.

Plan

Un exemple de base : la régression linéaire

- Un certain nombre de données expérimentales (x_i, y_i).
- On tente d'y voir un modèle de la forme $y = Ax + B$

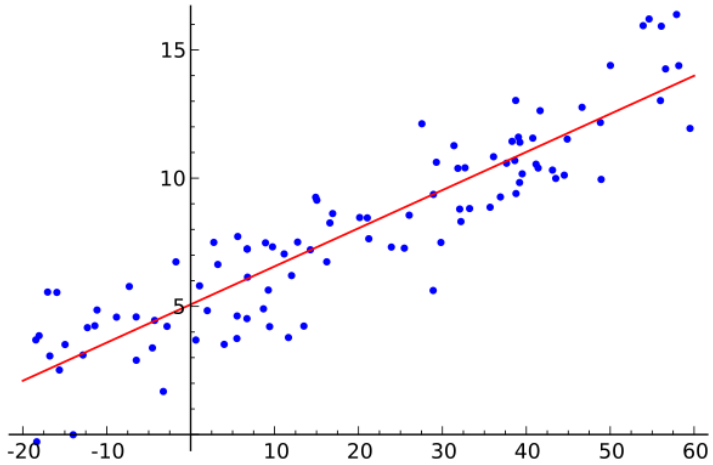
  Pourquoi ? raison théorique/aspect des mesures/test systématique



Un exemple de base : la régression linéaire

- Un certain nombre de données expérimentales (x_i, y_i).
- On tente d'y voir un modèle de la forme $y = Ax + B$

  Pourquoi ? raison théorique/aspect des mesures/test systématique



$\rightarrow$ obtenir un modèle = obtenir des valeurs de A et B convenables

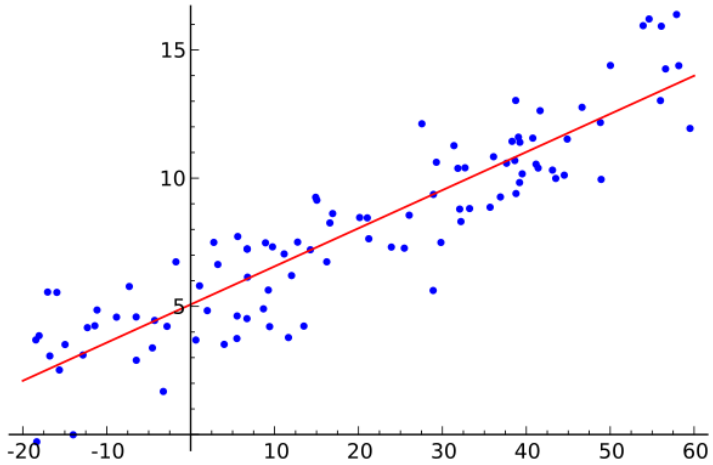
$\rightarrow$ quantifier la qualité du modèle

$\rightarrow$ si ce modèle semble pertinent : prévoir d'autres y à partir de x.

Un exemple de base : la régression linéaire

- Un certain nombre de données expérimentales (x_i, y_i).
- On tente d'y voir un modèle de la forme $y = Ax + B$

  Pourquoi ? raison théorique/aspect des mesures/test systématique



$\rightarrow$ obtenir un modèle = obtenir des valeurs de A et B convenables

$\rightarrow$ quantifier la qualité du modèle

$\rightarrow$ si ce modèle semble pertinent : prévoir d'autres y à partir de x.

Ici problème mathématiquement qualifié : recherche d'une distance minimale par projection, calcul d'un "coefficient de corrélation".

L'apprentissage supervisé

Forme générale du problème

Mathématiquement parlant, l'apprentissage supervisé a toujours la forme de la recherche d'une fonction $y = f(x)$ à partir de données étiquetées (x_i, y_i).

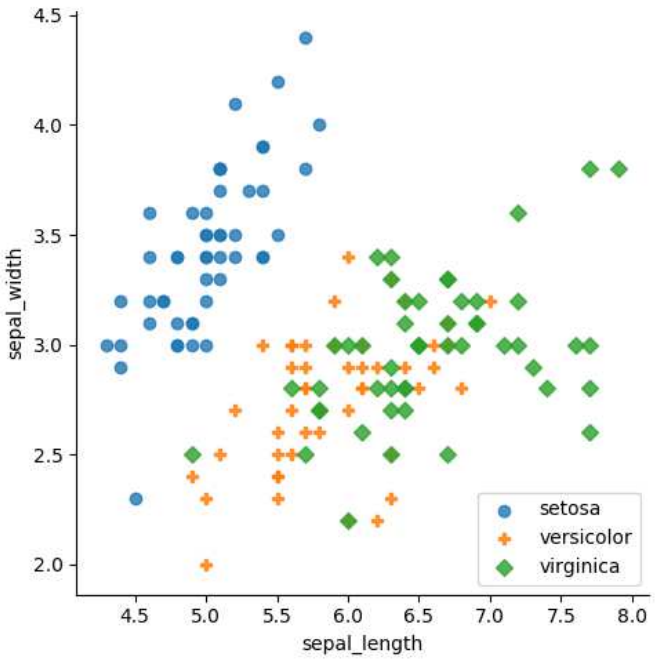
- les x_i sont les "observations" connues dans l'espace des données
- à chaque observation correspond l'"étiquette" y_i
- le modèle f est solution d'un problème d'optimisation : le meilleur
  selon un certain critère de proximité numérique
  dans une certaine famille de modèles

Les vecteurs x peuvent avoir de nombreuses composantes les "caractéristiques" (features).

Les images y peuvent être discrètes (problème de classification). Par exemple pour des végétaux $y_i = 4 \rightarrow$ "fargesia murielae simba".

Exemple : le problème de la classification

Ce sont les données d'apprentissage qu'on représente ici : c'est une série d'observations botaniques.

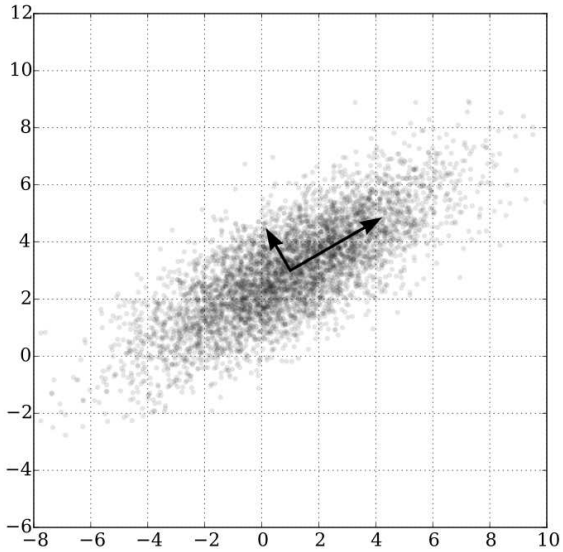


Ensuite, pour une nouvelle observation, il faudra être capable de proposer une prédiction, le plus proche possible des données connues.
Nous verrons notamment sur ce thème l'algorithme des k-plus proches voisins.

Plan

L'apprentissage non supervisé

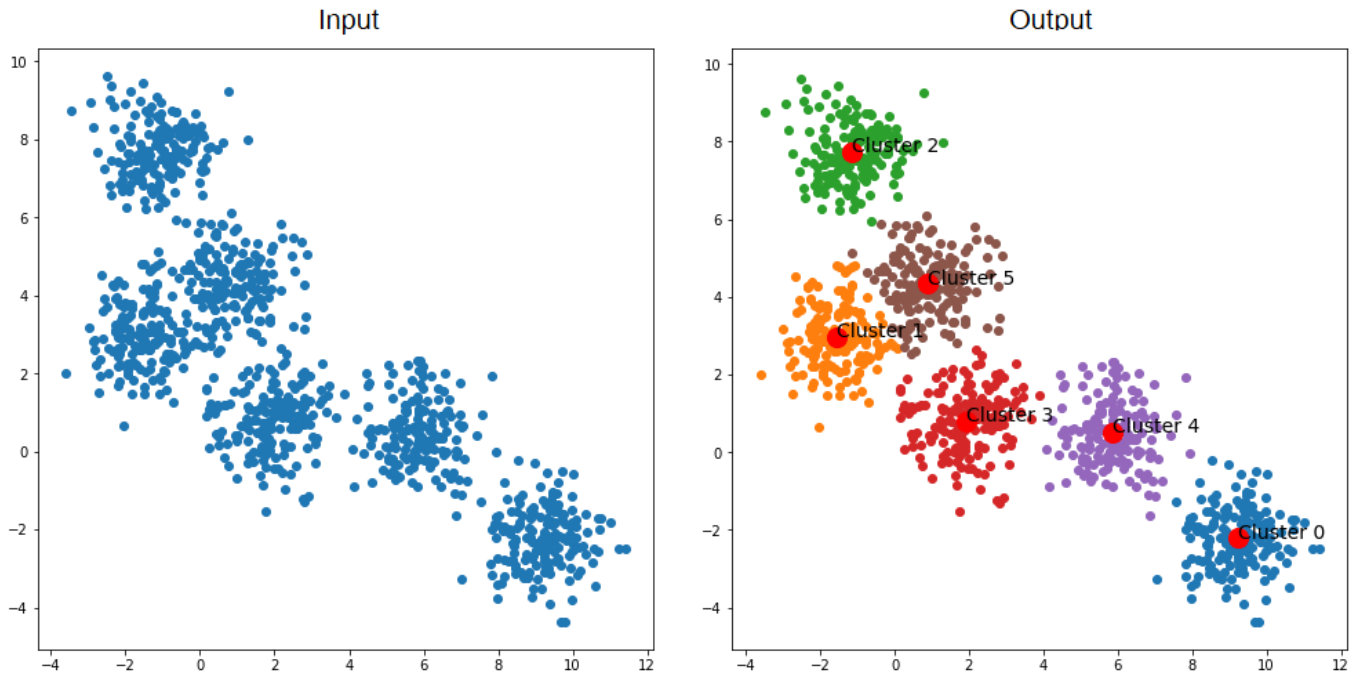
Il y a des techniques de réduction de dimensions



Ainsi l'analyse en composantes principales permet d'extraire les "coordonnées les plus significatives" d'un paquet de données.
Il faut imaginer qu'il y a en plus de nombreuses autres dimensions, mais avec une dispersion moins forte encore.

L'apprentissage non supervisé

Il y a également le "partitionnement" ("clustering") pour scinder les données en sous-groupes pertinents.



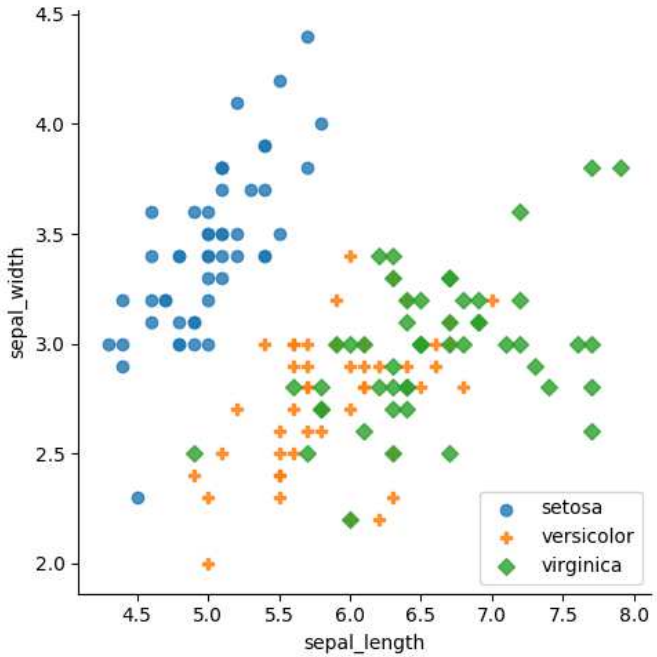
Ici on a pris un cas où les "clusters" se voient bien, ce qui est rarement le cas en pratique.

Ne pas confondre avec le problème de classification : ici les groupes ne sont pas connus. On verra un autre algorithme, celui des k-moyennes.

Plan

Le problème de la classification

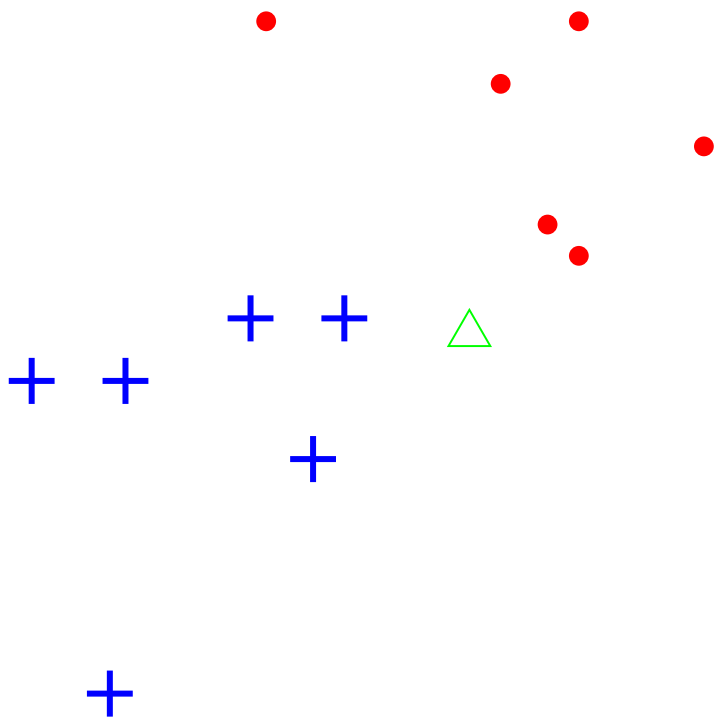
Les données d'apprentissage sont de la forme (x_i, y_i), avec x_i à valeurs dans l'espace n-dimensionnel $\mathbb{R}^n$ des caractéristiques. Les étiquettes y_i sont des entiers : 0/1 pour une "classification binaire", ou des entiers de 1 à n (classification multi-classes).



Plan

Algorithme des k plus proches voisins

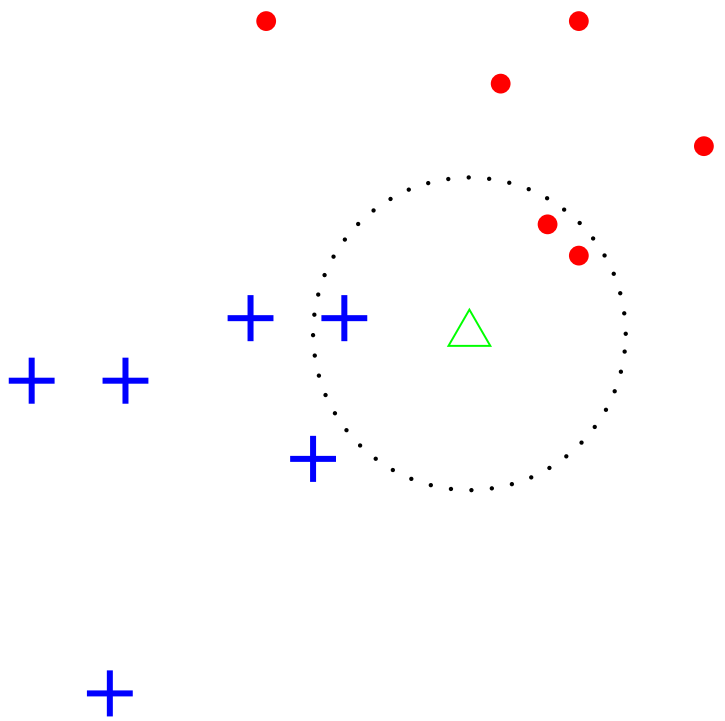
On prend k impair (1,3,5...) et on cherche parmi les données d'apprentissage, les k les plus proches.



Algorithme des k plus proches voisins

On prend k impair (1,3,5...) et on cherche parmi les données d'apprentissage, les k les plus proches.

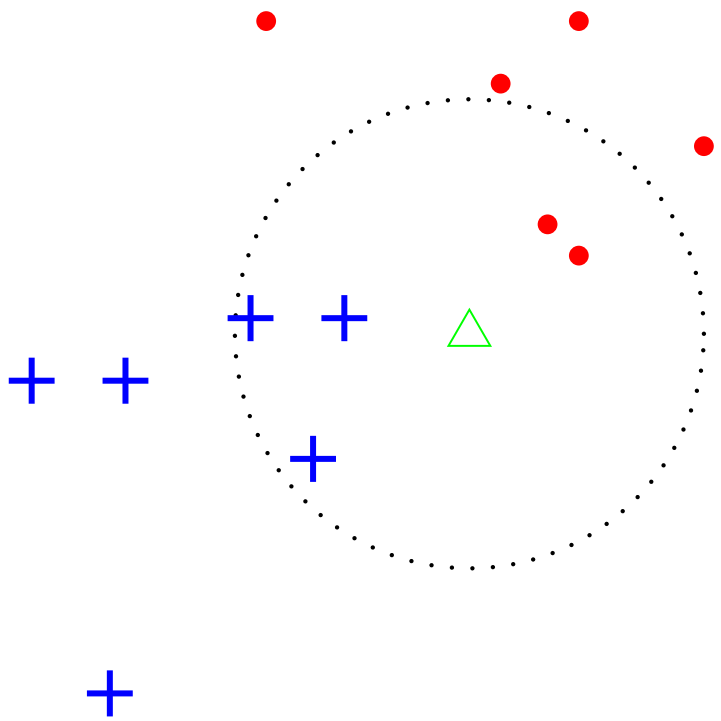
Effet pour $k = 3$: classé rouge



Algorithme des k plus proches voisins

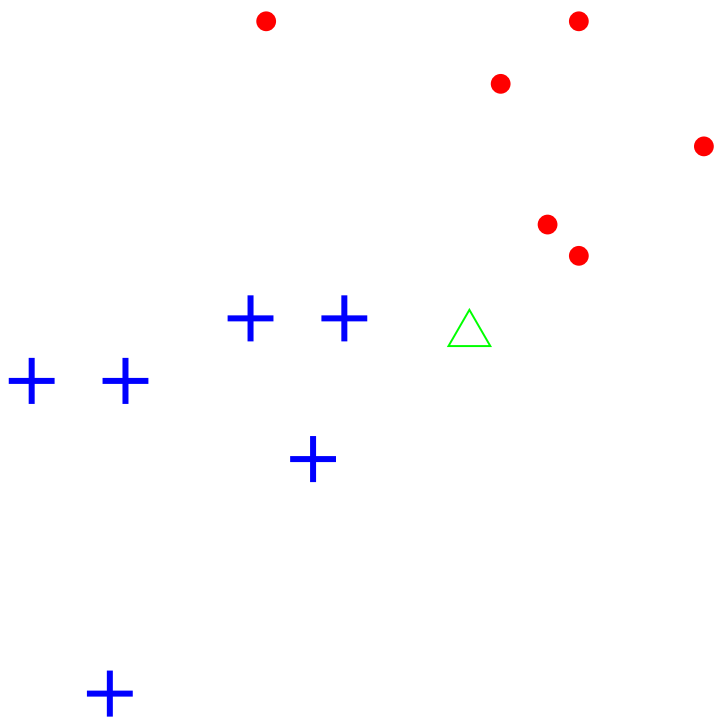
On prend k impair (1,3,5...) et on cherche parmi les données d'apprentissage, les k les plus proches.

Effet pour $k = 5$: classé bleu



Algorithme des k plus proches voisins

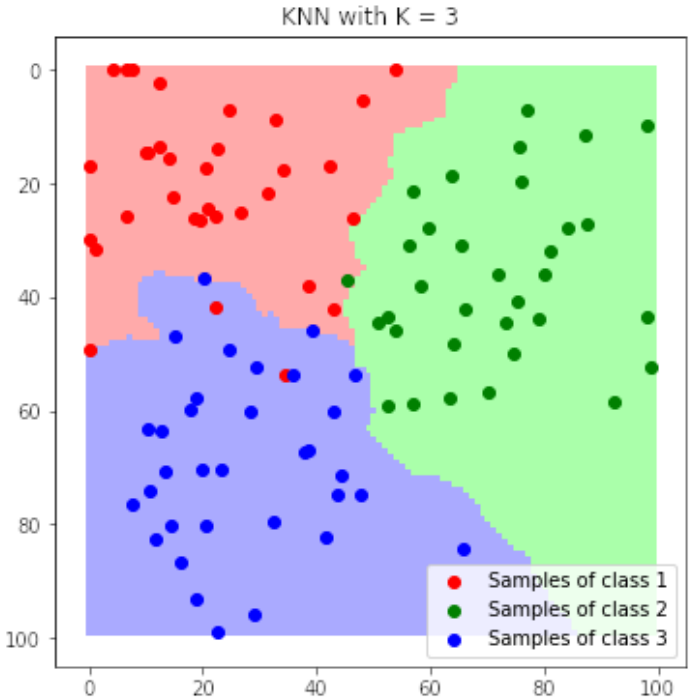
On prend k impair (1,3,5...) et on cherche parmi les données d'apprentissage, les k les plus proches.

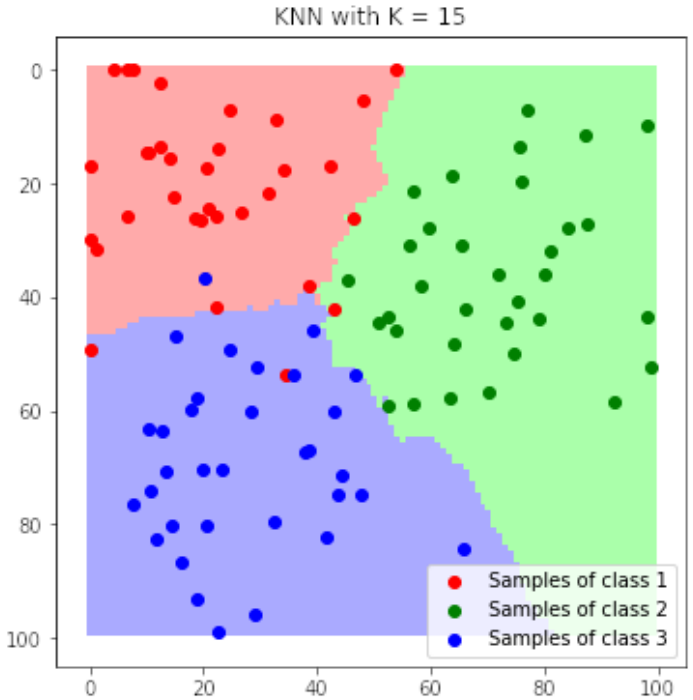


Evidemment on a pris un point particulièrement discutable. Beaucoup d'autres se classeront sans difficulté.

Visualisation : frontière de décision

On peut faire cette visualisation pour un problème à deux dimensions seulement





Comment décider quel est le "meilleur choix" ?

Plan

Evaluer le résultat

Il 's'agit de façon générale d'évaluer un algorithme de classification. Pour tester si l'algorithme est pertinent on peut partir d'un large panel de données étiquetées
- en consacrer une petite partie à l'apprentissage
- et la plus grosse partie à l'évaluation

Déf - Matrice de confusion pour évaluer une classification

Coefficient i, j de la matrice = le nombre d'échantillons dont la classe véritable est i qui ont été classées j.

Plus cette matrice est proche d'être diagonale mieux c'est.

$$\begin{pmatrix} 16 & 0 & 0 \\ 0 & 9 & 1 \\ 0 & 1 & 11 \end{pmatrix}$$

Exemple : évaluer une classification binaire

Par exemple, pour évaluer un test médical (problème de classification binaire)

	0	1
0	vrais négatifs	faux négatifs
1	faux positifs	vrais positifs

On définit deux indicateurs numériques (programme de SI)

- sensibilité : capacité à détecter la maladie qd elle est présente

$$s = \frac{\mathrm{VP}}{\mathrm{Malades}} = \frac{\mathrm{VP}}{\mathrm{VP} + \mathrm{FN}}$$

- spécificité : capacité à écarter la maladie qd elle est absente

$$s = \frac{\mathrm{VN}}{\mathrm{Non-malades}} = \frac{\mathrm{VN}}{\mathrm{VN} + \mathrm{FP}}$$

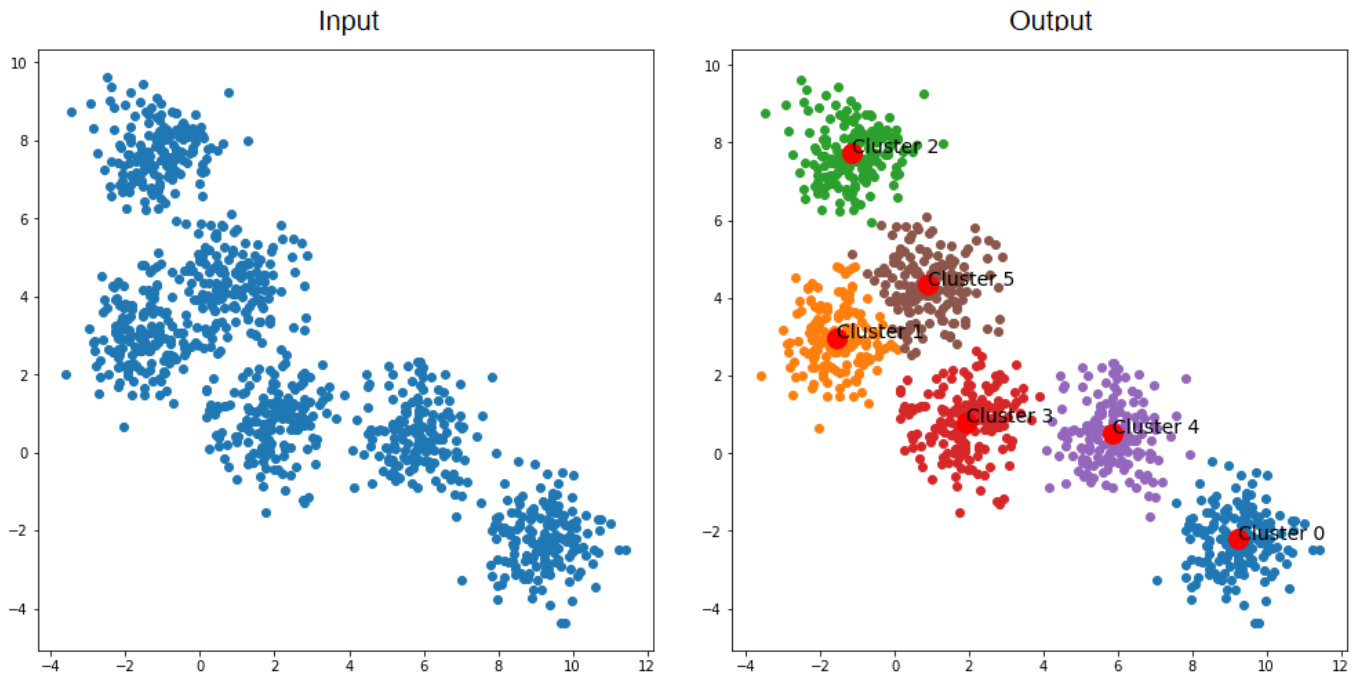
Avantages et faiblesses

- facile à implémenter mais lent dans sa version de base (détermination des k plus proches)
- il faut se demander comment régler k (souvent 3 ou 5)
- mais surtout : quel choix de fonction distance ?
- par suite ce n'est pas très adapté quand l'espace des caractéristiques a beaucoup de dimensions
- pertinence d'émettre un vote à la majorité quand les effectifs sont très différents
- d'où l'idée parfois de pondérer les votes (par une fonction de la distance, par les effectifs)...

Plan

Le problème du partitionnement

Il s'agit de scinder les données en sous-groupes pertinents, mais on n'a pas d'information a priori, juste les données brutes.



Ici on a pris un cas où les "clusters" se voient bien, ce qui est rarement le cas en pratique.

Le problème du partitionnement

- Données multidimensionnelles ($\mathbb{R}^d$)
- à répartir en classes (ou groupes) aussi homogènes que possible
- selon des critères de distance mutuelle (distance ordinaire de $\mathbb{R}^d$)
- sans information a priori (non supervisé).

Nombre de classes a priori inconnu

Même le nombre de groupes pertinents est inconnu : on peut tenter une valeur k, une autre,... et chercher ce qui est mieux.

Dans un premier temps on le fixe à une valeur donnée k

Une quantité à minimiser

On cherche à partitionner les données sous la forme $(S_1, \ldots, S_k)$ en minimisant les carrés des distances intra-blocs : on définit la distorsion

$$D = \sum_{i=1}^{k} \frac{1}{|S_i|} \sum_{(x,y)\in S_i^2} \|x - y\|^2$$

Une quantité à minimiser

On cherche à partitionner les données sous la forme $(S_1, \ldots, S_k)$ en minimisant les carrés des distances intra-blocs : on définit la distorsion

$$\boxed{D = \sum_{i=1}^{k} \frac{1}{|S_i|} \sum_{(x,y)\in S_i^2} \|x - y\|^2}$$

Argument classique de géométrie : introduire l'isobarycentre μ_i des éléments de S_i

$$\sum_{(x,y)\in S_i^2} \|x - y\|^2 = \sum_{(x,y)\in S_i^2} \|(x - \mu_i) + (\mu_i - y)\|^2$$

en développant ce carré, les produits scalaires disparaissent et

$$\sum_{(x,y)\in S_i^2} \|x - y\|^2 = 2|S_i| \sum_{(x,y)\in S_i^2} \|x - \mu_i\|^2$$

Une quantité à minimiser

On cherche à partitionner les données sous la forme $(S_1, \ldots, S_k)$ en minimisant les carrés des distances intra-blocs : on définit la distorsion

$$D = \sum_{i=1}^{k} \frac{1}{|S_i|} \sum_{(x,y)\in S_i^2} \|x - y\|^2$$

On arrive à

$$D = 2 \sum_{i=1}^{k} \sum_{x\in S_i} \|x - \mu_i\|^2$$

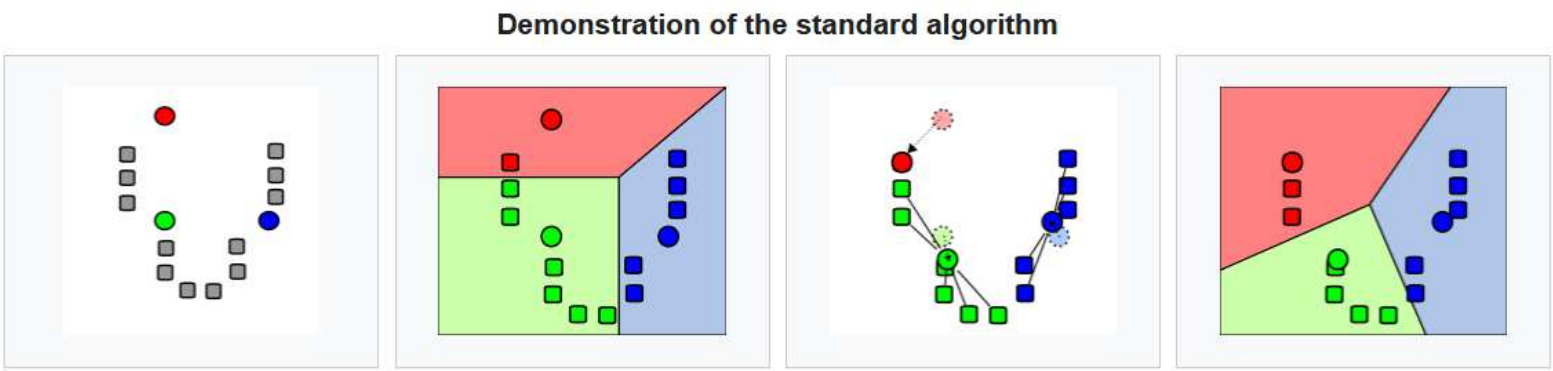
qu'on peut également interpréter en termes de variance.

On va voir un algorithme qui permet de trouver des minima locaux (pas toujours globaux) de la distorsion.

Plan

Algorithme des k-moyennes

- Initialisation : choix de k données particulières (voir plus tard) qui seront nos "centres"
- Succession d'étapes
  - affectation de chaque donnée à un centre "optimal"
  - recentrage en calculant de nouveaux isobarycentres
- et ceci jusqu'à ce que la situation n'évolue plus : pas de réaffectation.



Quelles garanties ?

On peut montrer que l'algorithme converge (du moins quand on travaille effectivement avec la norme euclidienne). Mais

- le nombre d'étapes est souvent limité, mais parfois vraiment long

- la complexité globale reste élevée de toute façon

- surtout on tombe couramment sur des extrema locaux peu intéressants (cf TP).

Quelles garanties ?

On peut montrer que l'algorithme converge (du moins quand on travaille effectivement avec la norme euclidienne). Mais

- le nombre d'étapes est souvent limité, mais parfois vraiment long

- la complexité globale reste élevée de toute façon

- surtout on tombe couramment sur des extrema locaux peu intéressants (cf TP).

Il y a différentes pistes d'amélioration un peu lourdes, comme faire tourner plusieurs fois l'algorithme et prendre la distorsion la plus faible...

Plan

Stratégies d'initialisation

Vous n'avez pas à les connaître...

- le plus simple (Forgy) : prendre k données au hasard

- un autre départ aléatoire : affecter chacun des points au hasard et attaquer par l'étape de recentrage.

En termes qualitatifs on voit quels défauts peuvent présenter ces choix.

Stratégies d'initialisation

Vous n'avez pas à les connaître...

- le plus simple (Forgy) : prendre k données au hasard

- un autre départ aléatoire : affecter chacun des points au hasard et attaquer par l'étape de recentrage.

En termes qualitatifs on voit quels défauts peuvent présenter ces choix.

Etudié en TP : l'algorithme k-means++ (2007) propose un choix initial adapté à la géométrie

Pour le TP : l'idée de k-means++

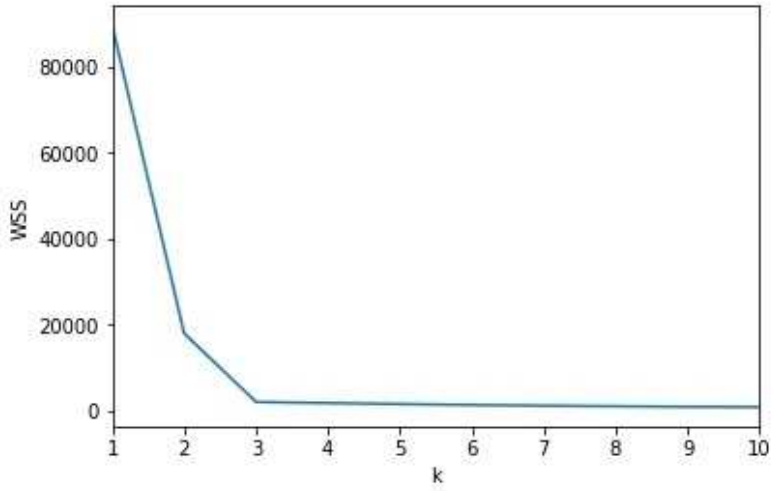
Pour initialiser,

- choisir une première donnée D1 au hasard
- choisir la seconde D2 en la cherchant plutôt loin de la première
  loi de probabilité pondérée par la distance au carré
- choisir la troisième en la cherchant plutôt loin des deux premières
  loi de probabilité : la plus petite distance au carré (avec D1 ou D2)
- etc

Plan

Trouver le bon k

Une approche naïve consiste à tester les valeurs de k les unes après les autres, à mesurer la distorsion.
La distorsion a tendance à diminuer, d'abord très fortement, puis très peu.
On peut utiliser la valeur de k qui provoque la séparation entre ces deux comportements ("méthode du coude" car le graphe fait un coude).



Après k=3, plus d'amélioration notable.

Il existe aussi des critères quantitatifs plus précis pour estimer la qualité du partitionnement et aider notamment à choisir le bon k : on verra en TP le coefficient de silhouette.