Formation Machine Learning

Premiers modèles supervisés

Giraud François-Marie

4 Juin 2019

ENI Service

Machine Learning

Regression Linéaire

Regression Linéaire

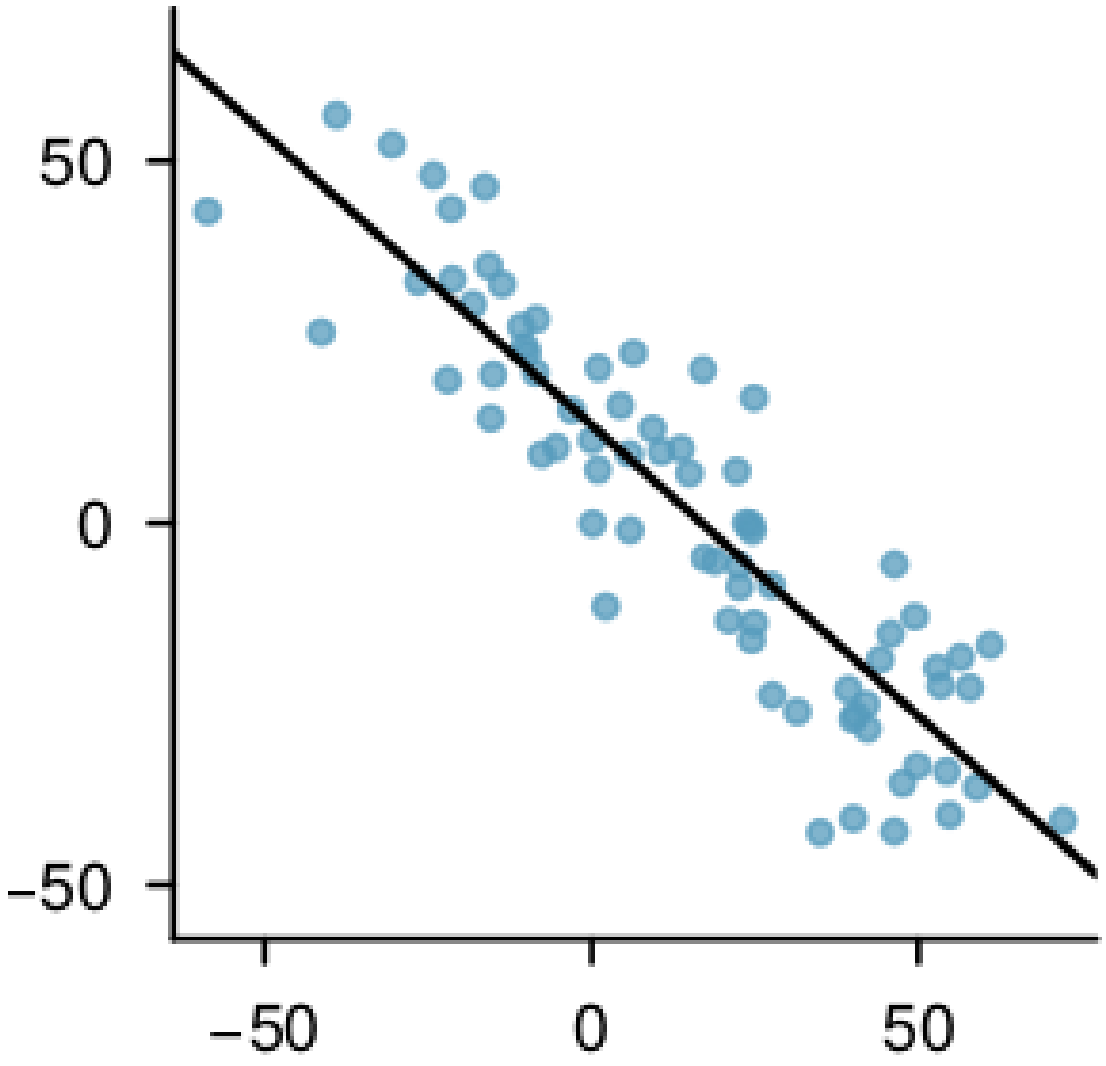
Définition du problème :

Soit $\{(x_i, y_i)\}_{i \in \mathbb{R}}$ un ensemble de données tel que $\forall i, x_i \in \mathbb{R}$ et $y_i \in \mathbb{R}$

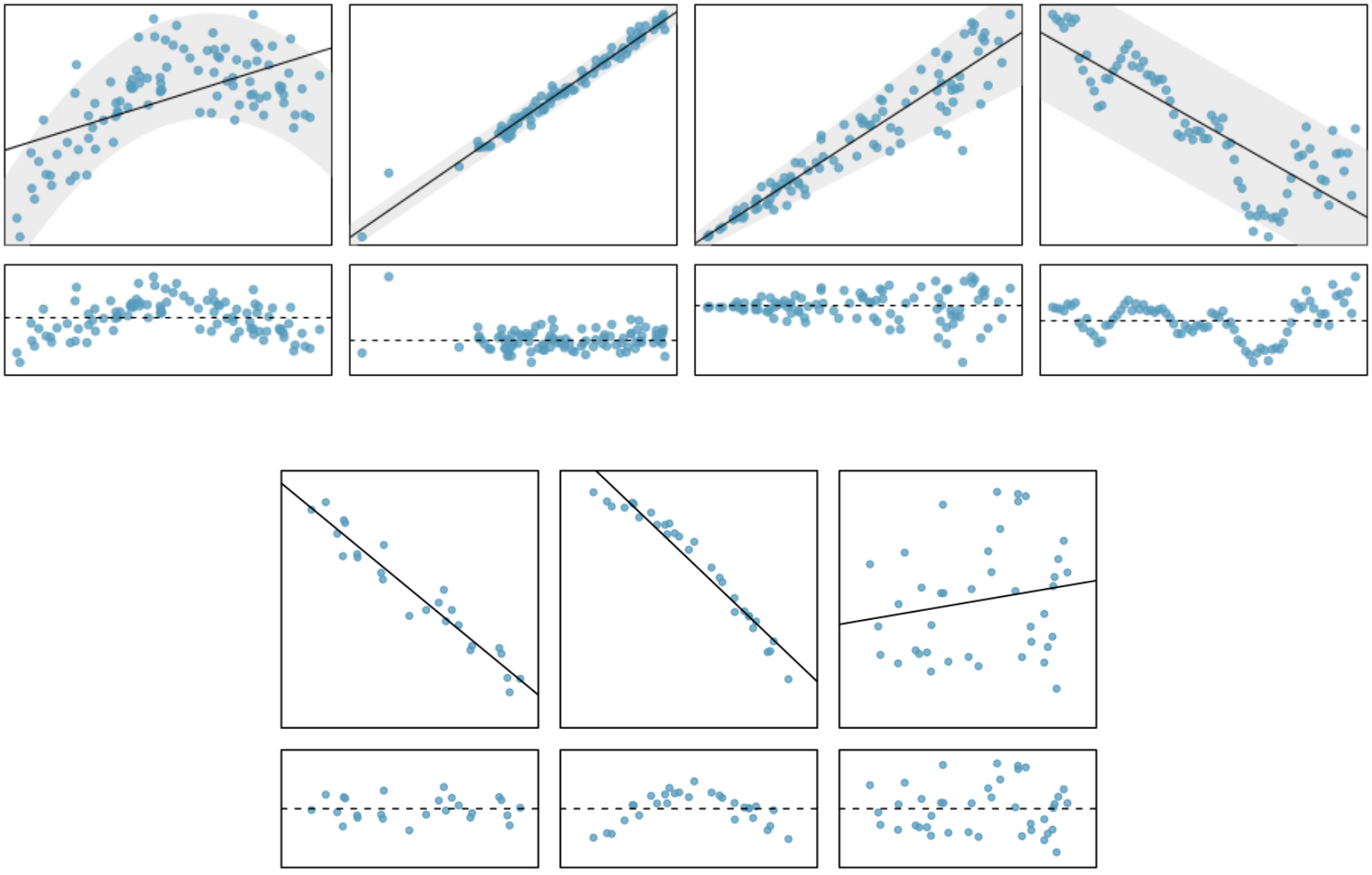
Trouver $\phi^*(x_i) = y_i^*$ telle que

$\forall i, y_i^* - y_i \to 0$
sous la contrainte que ϕ^* soit une fonction linéaire (affine)

Regression Linéaire



Regression Linéaire



Machine Learning

Fonction de Coût/Erreur

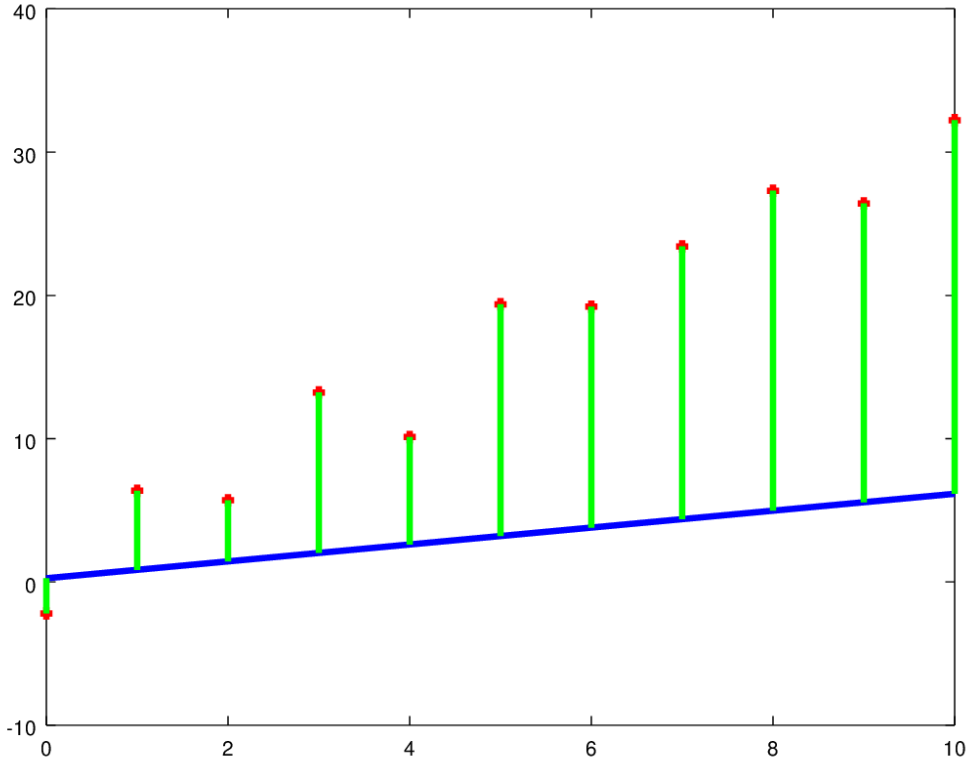
Fonction de Coût/Erreur

Erreur moyenne :

$$\frac{1}{n} \sum_{i=[1..n]} \sqrt{(y_i^* - y_i)^2}$$

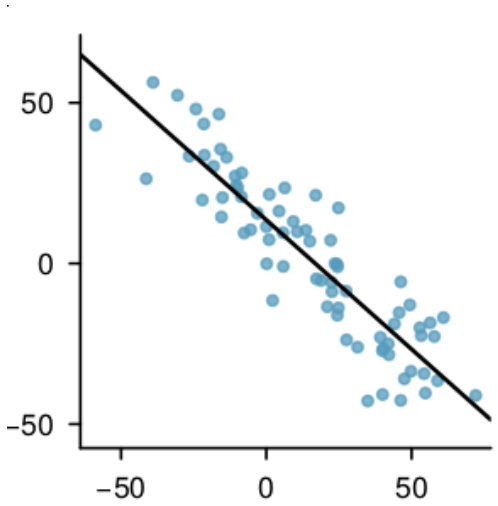
Critère des moindres carrés :

$$\frac{1}{n} \sum_{i=[1..n]} (y_i^* - y_i)^2$$



Fonction de Coût/Erreur

$$x = a.x + b$$



$$E_\Omega = \frac{1}{2n} \sum_{i=[1..n]} (y_i^* - y_i)^2$$

Machine Learning

Optimisation

Des solutions analytiques existent !

mais ...

Optimisation

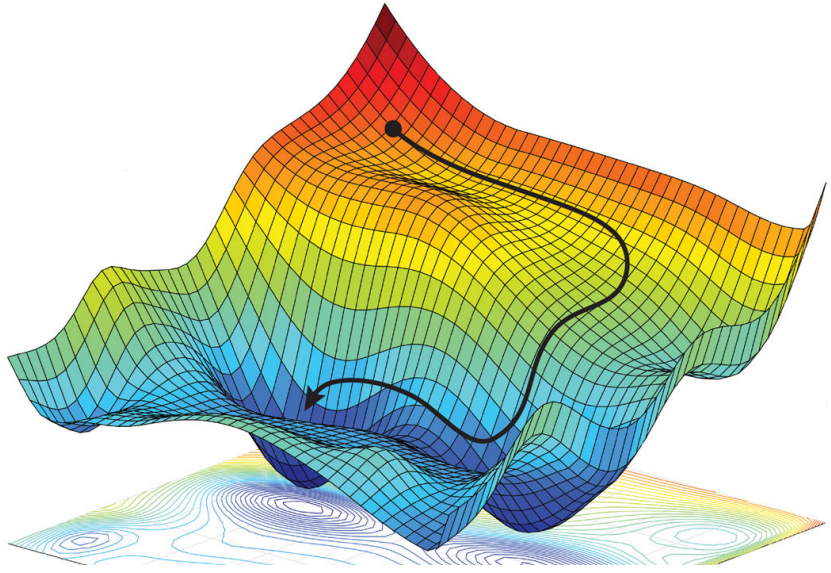
Calcul du gradient de l'erreur par rapport aux paramètres :

$$\frac{\partial Err}{\partial w_i}$$

Mise à jour :

$$w_i' = w_i - \gamma * grad$$

où : $0 < \gamma < 1$ (learning rate)



Optimisation

1 - initialisation aléatoire du modèle
2 - Tant que(critère arret == 0)

- Selection aléatoire d'un **batch** de données
- **Forward** : Passe avant du **batch** dans le modèle
- Calcul de l'erreur par rapport aux sorties attendues
- **Backward** : Rétropropagation du gradient de l'erreur en fonction des paramèrtres dans le modèle (mise à jour du modèle)
- Calcul critère arret

Machine Learning

Gradient de l'erreur

Gradient et mise à jour

$$\boxed{y = a.x + b}$$

$$E_\Omega = \frac{1}{2n} \sum_{i=[1..n]} (y_i^* - y_i)^2$$
$$E_\Omega = \frac{1}{2n} \sum_{i=[1..n]} (y_i^* - (a.x_i + b))^2$$
...
$$\frac{\partial E_\Omega}{\partial a} = \frac{1}{n} \sum_{i=[1..n]} (a.x_i + b - y_i^*).x$$
$$\frac{\partial E_\Omega}{\partial b} = \frac{1}{n} \sum_{i=[1..n]} (a.x_i + b - y_i^*)$$

$$\boxed{U^{2\,\prime} = 2U' * U}$$

M.A.J :

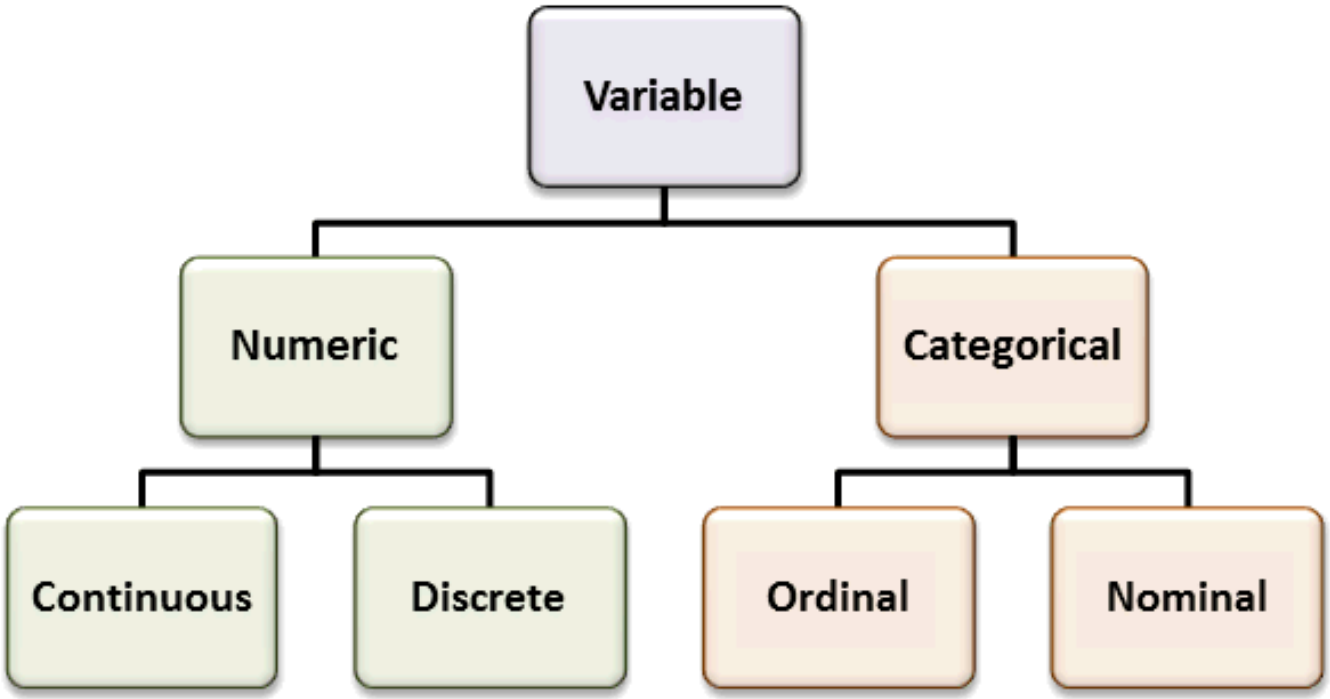
$$a \leftarrow a - \gamma . \frac{\partial E_\Omega}{\partial a}$$
$$b \leftarrow b - \gamma . \frac{\partial E_\Omega}{\partial b}$$

où $1 > \gamma > 0$ (learning rate)

Machine Learning

Régression Logistique

Régression Logistique



Régression Logistique

Régression Logistique

001.ak47

002.american-flag

003.backpack

004.baseball-bat

005.baseball-glove

006.basketball-hoop

007.bat

008.bathtub

009.bear

010.beer-mug

011.billiards

012.binoculars

Régression Logistique

$\approx$ Distance entre la sortie et la cible ?

Sortie :

0.0	0.1	0.4	0.0	0.0	0.2	0.1	0.0	0.2	0.0

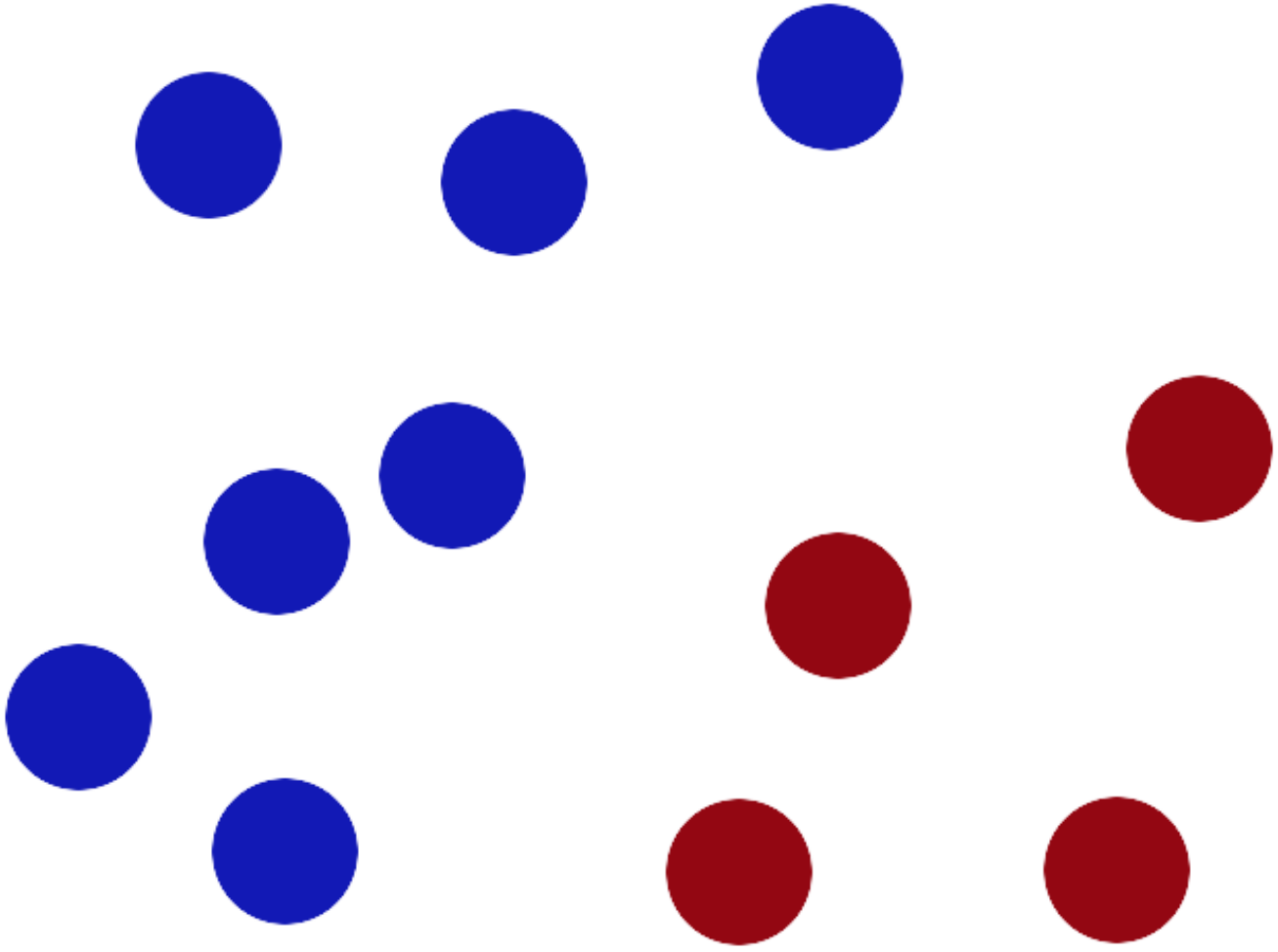
Cible :

0.0	0.0	**1.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0

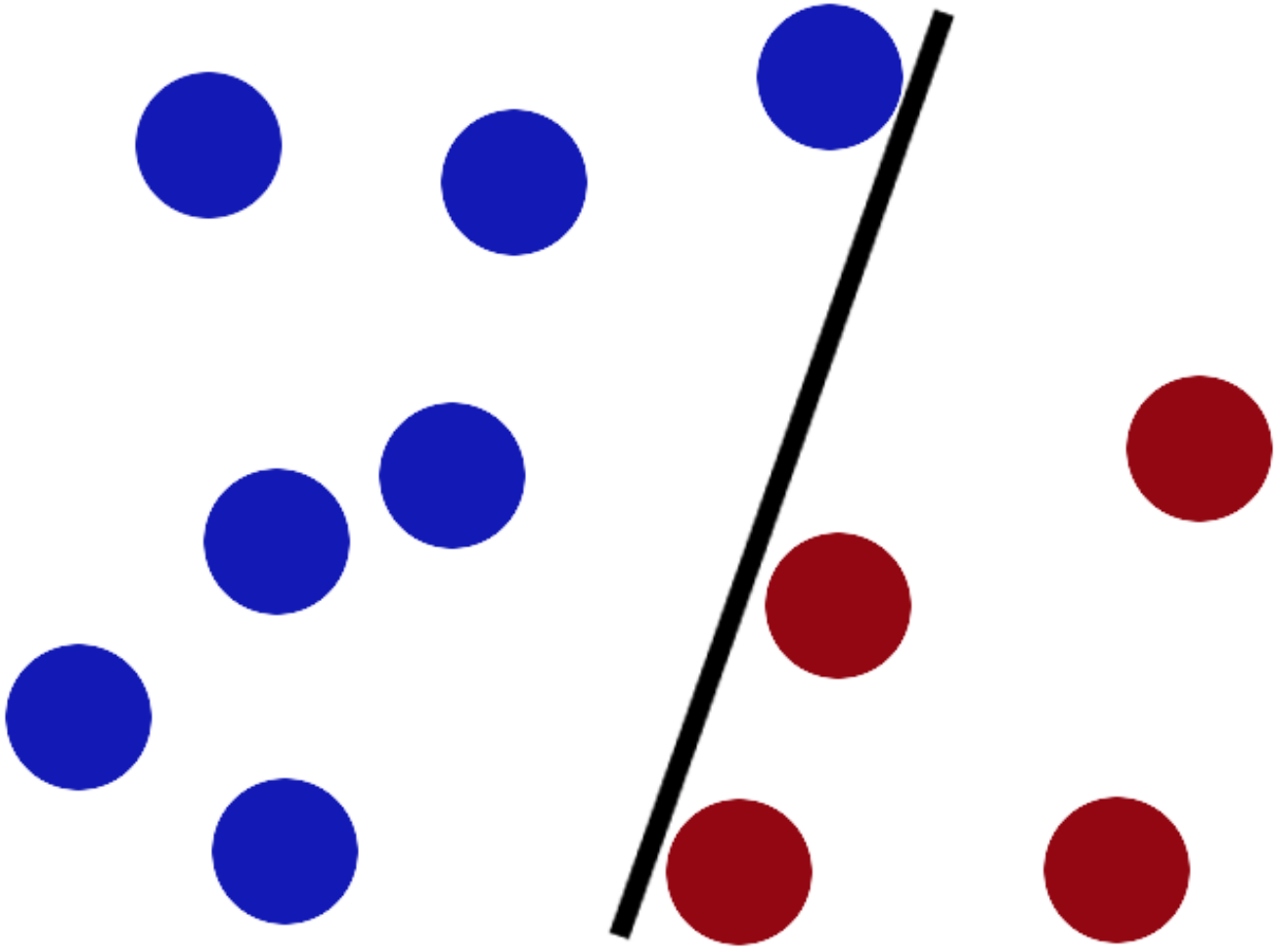
Machine Learning

Support Vector Machine

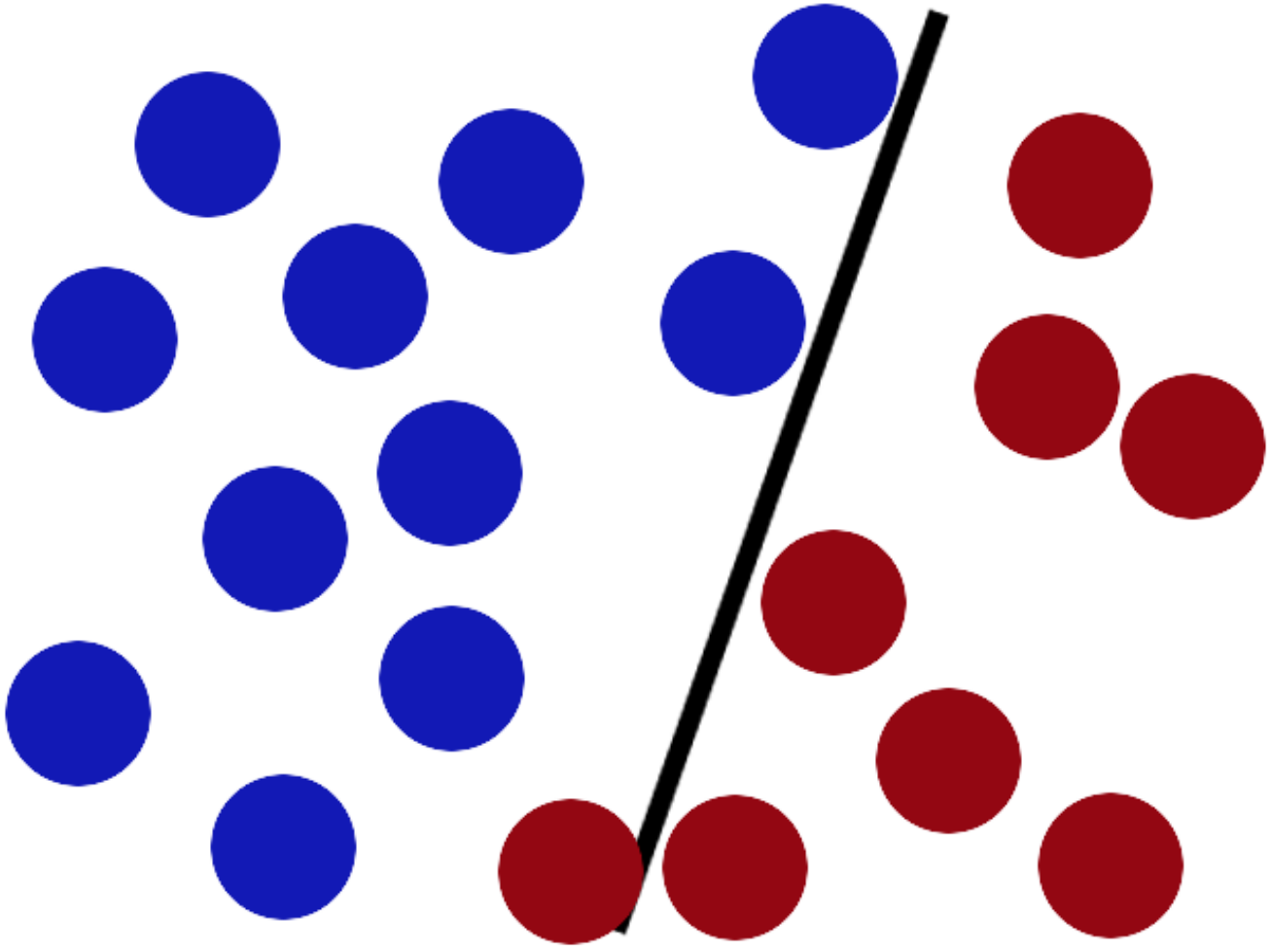
Support Vector Machine



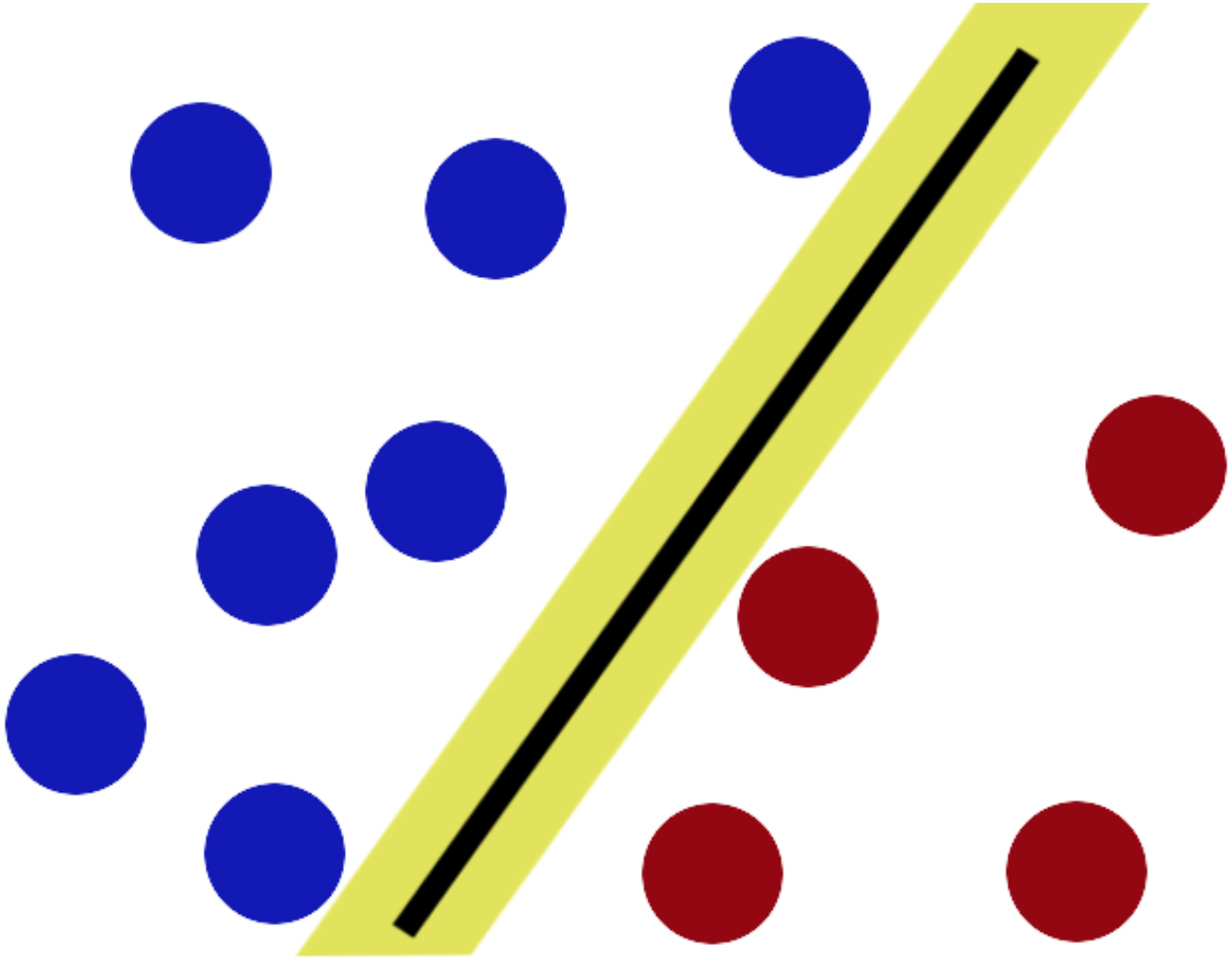
Support Vector Machine



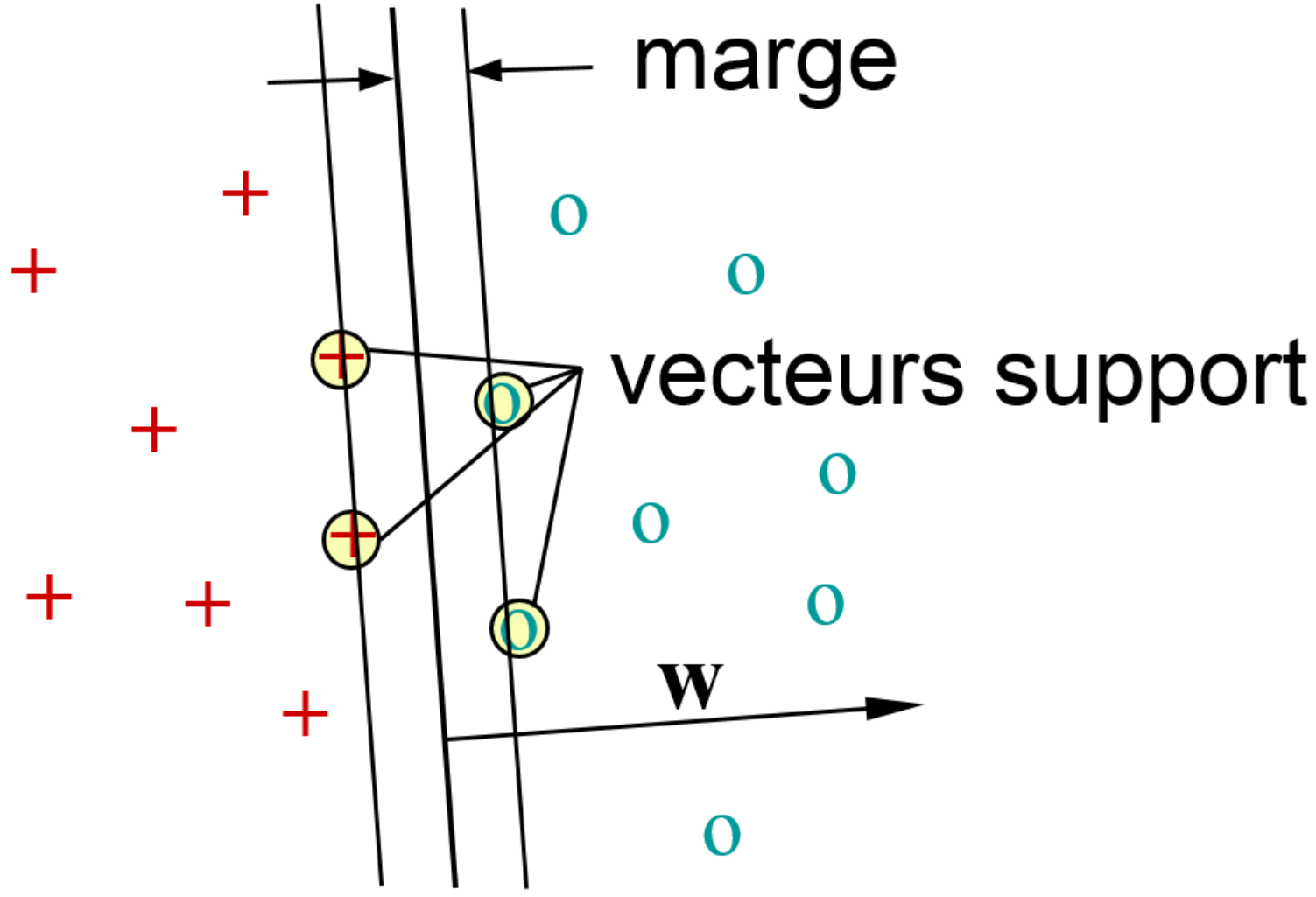
Support Vector Machine



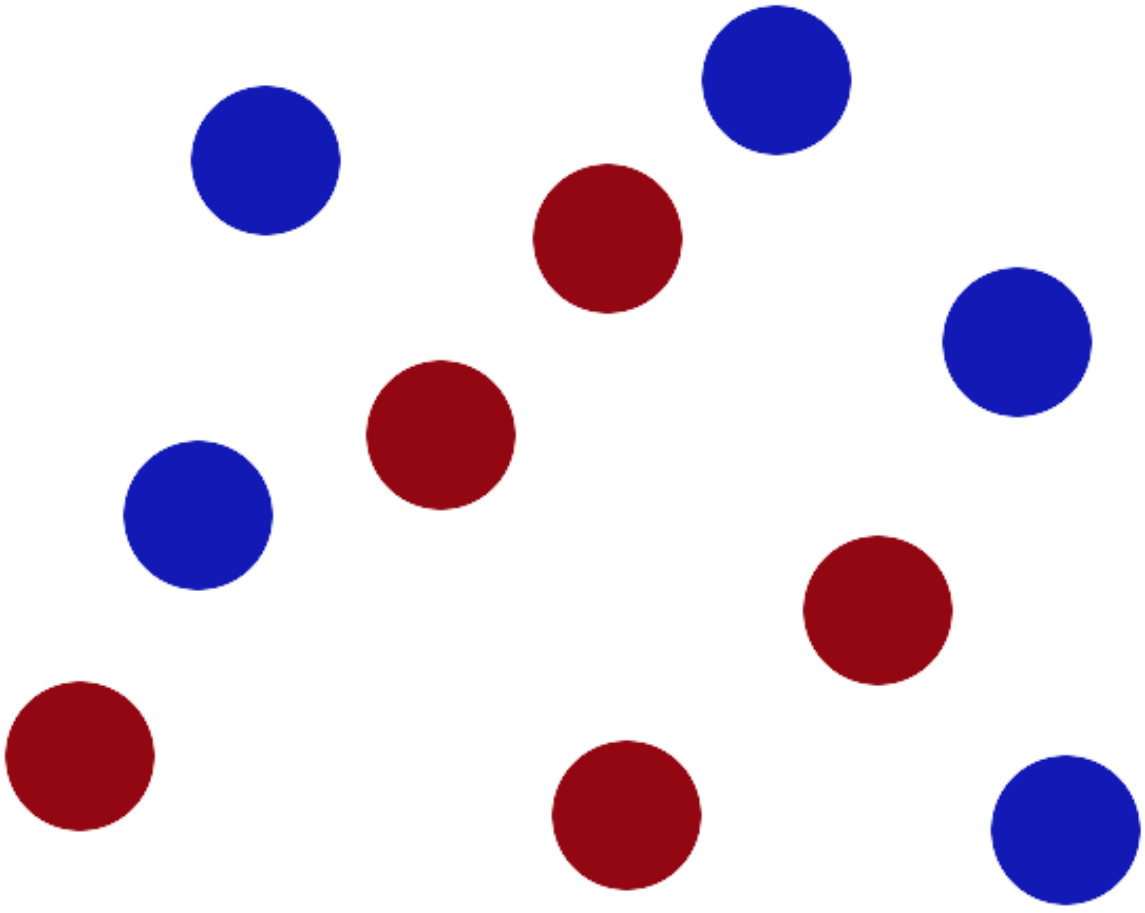
Support Vector Machine



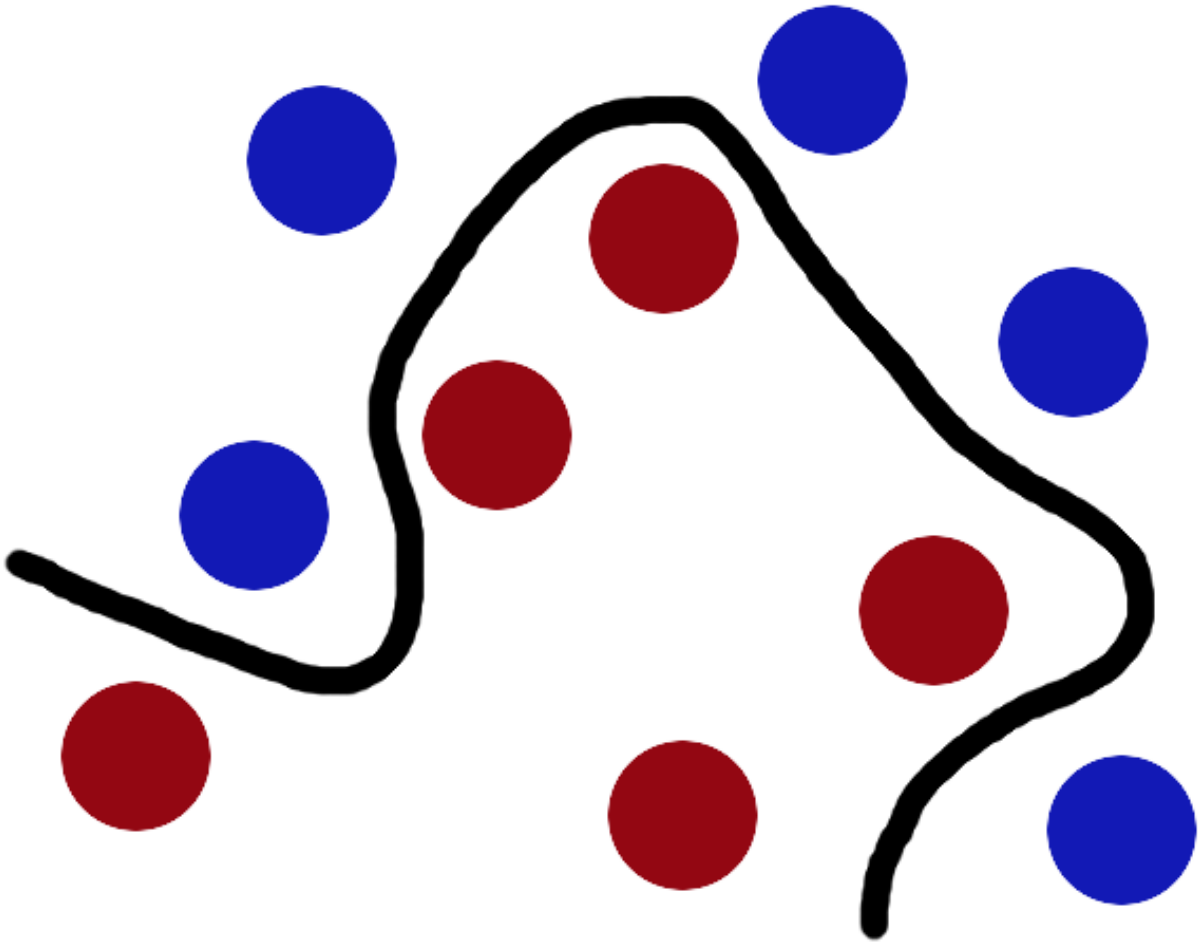
Support Vector Machine



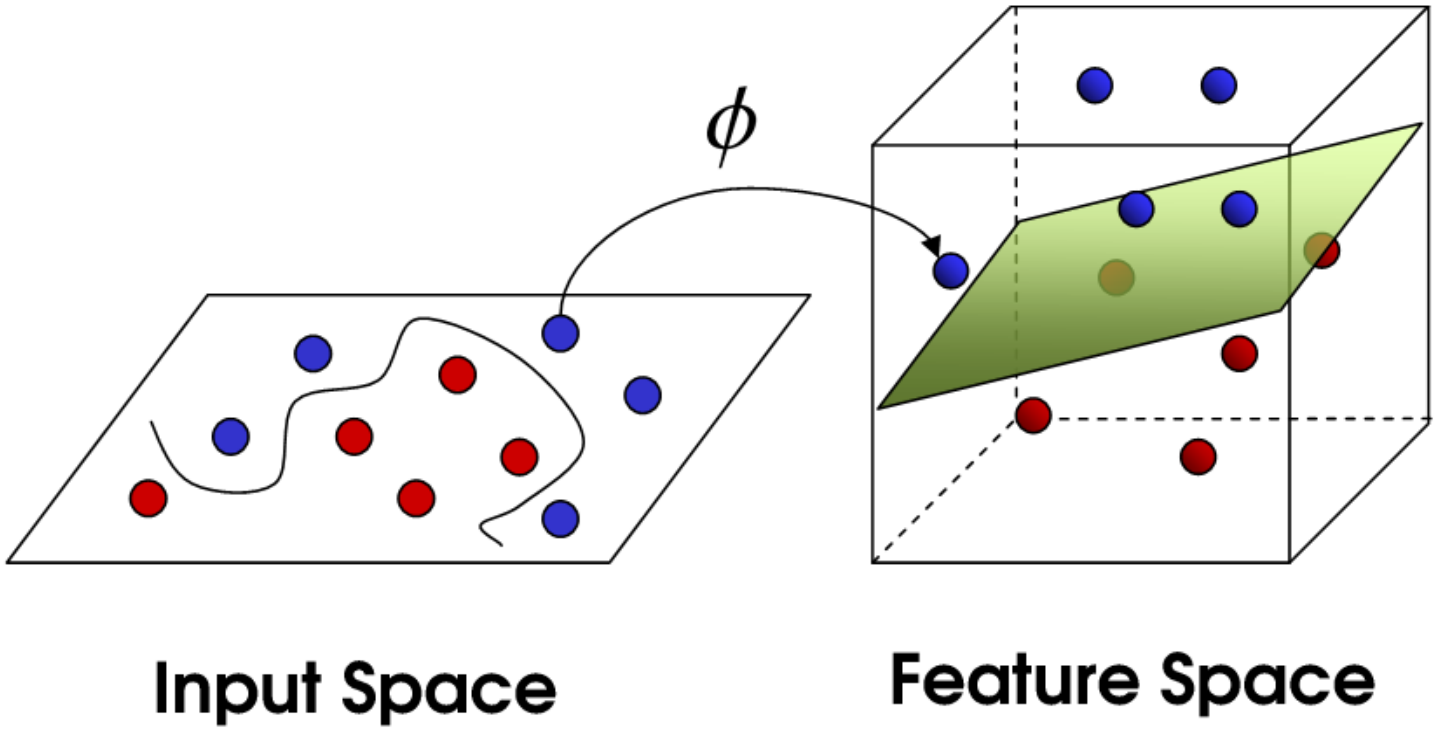
Support Vector Machine



Support Vector Machine



Support Vector Machine



Support Vector Machine

Petite vidéo d'explication des méthodes à noyaux (Kernel SVM)

Support Vector Machine

Généralisation à un problème de régression logistique à $K > 2$ classes :

- One Vs All : K modèles. Agréagation par meilleur score.
- One Vs One : $\frac{K(K-1)}{2}$ modèles. Vote majoritaire.

QUESTIONS ?

Arbres de décision

Module 5

Objectifs

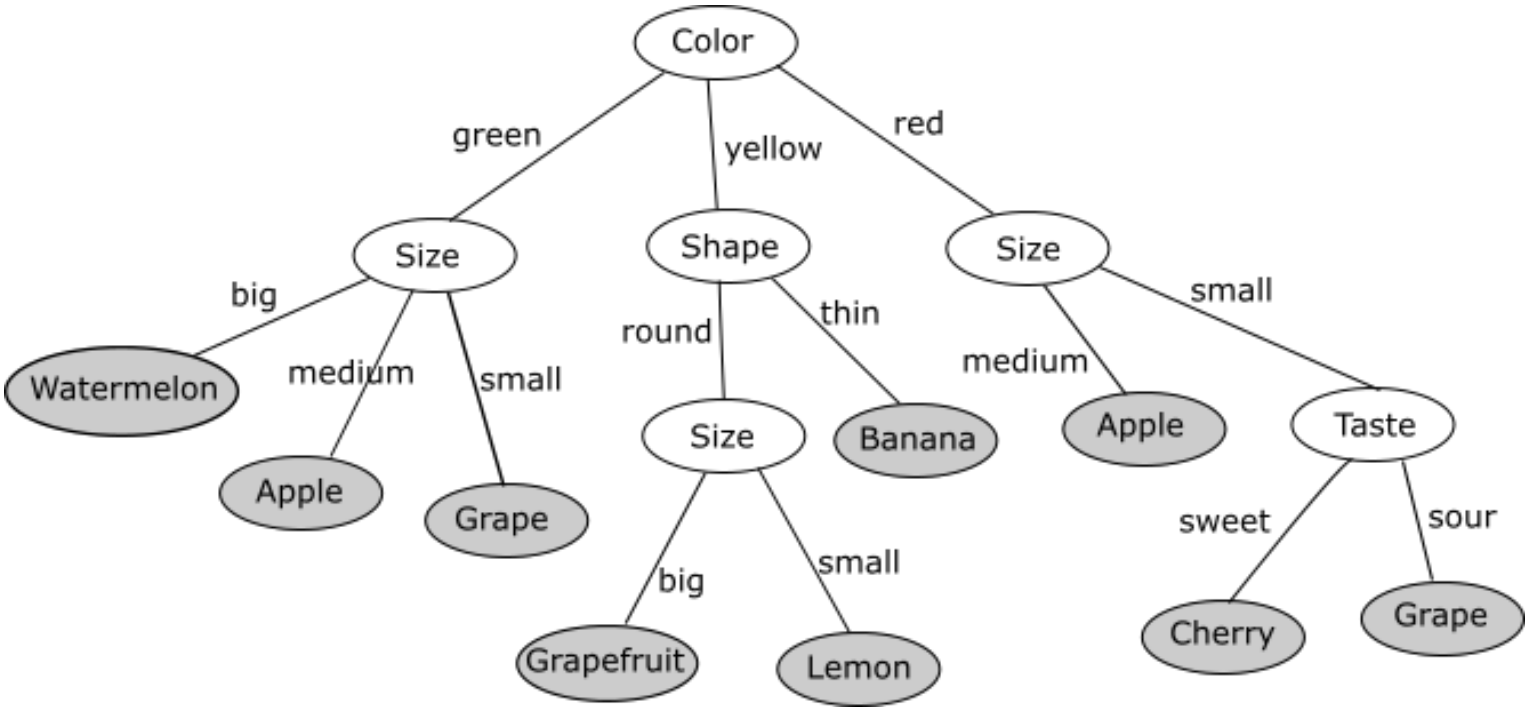
Objectifs

- construire un arbre de décision aussi bien pour la régression que pour la classification
- combiner plusieurs arbres efficacement avec Random Forest

Arbres de décision

Introduction

Modèle de classification ou regréssion qui classe un input dans une de ses feuilles pour rendre sa prédiction :



Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données
- leur apprentissage est hautement parallèlisable

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données
- leur apprentissage est hautement parallèlisable

$\rightarrow$ Couteau-suisse du machine learning tabulaire.

Désavantages

- peuvent overfit les données, mais l'ensembling résoud ce problème

Désavantages

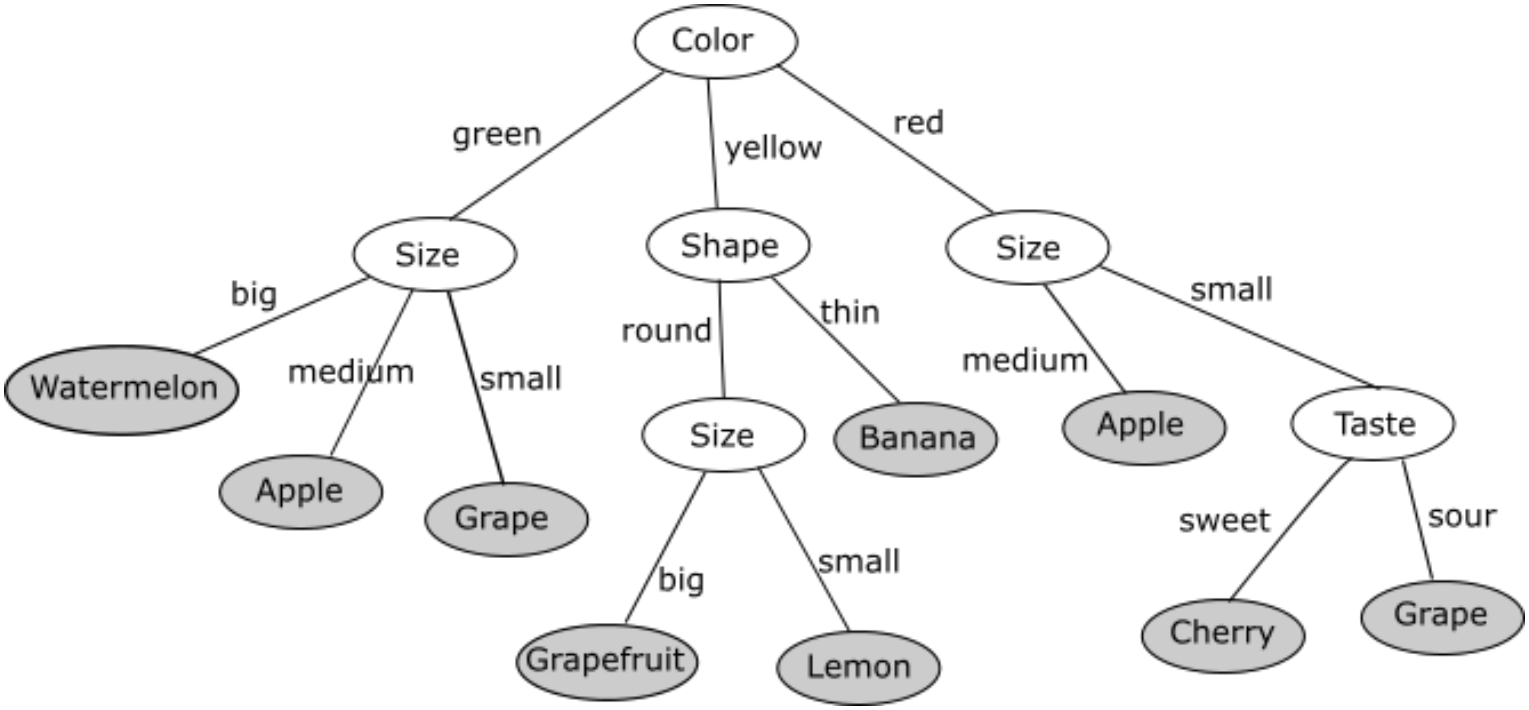
- peuvent overfit les données, mais l'ensembling résoud ce problème
- sont sensibles aux déséquilibres de classe

Désavantages

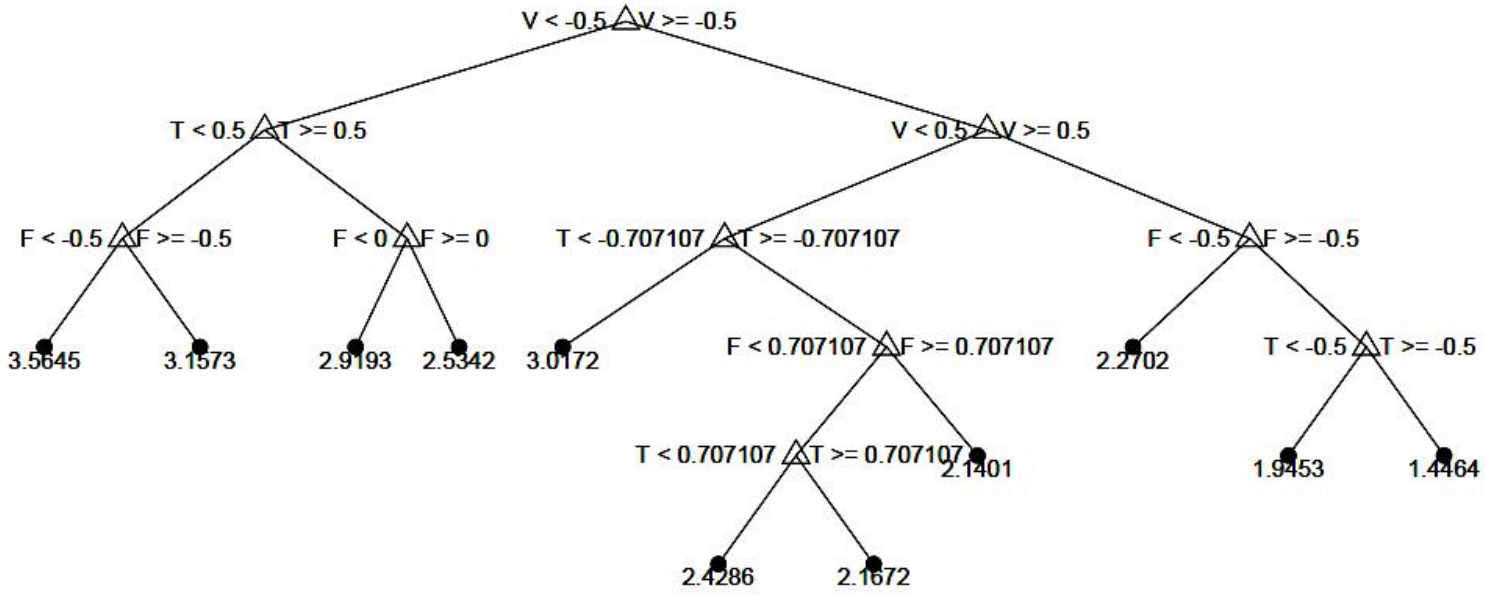
- peuvent overfit les données, mais l'ensembling résoud ce problème
- sont sensibles aux déséquilibres de classe

$\rightarrow$ Si les classes ne sont pas équilibrées, peut-être les resampler.

Arbres de classification



Arbres de régression



Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire
  - sinon, créer deux branches à partir du nœud non traité pour répartir les instances dans deux nouveaux nœuds

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire
  - sinon, créer deux branches à partir du nœud non traité pour répartir les instances dans deux nouveaux nœuds

Conditions de feuilles finales : contient n_{min} éléments, est déjà à profondeur p_{max}, splitterait sans décroître assez l'entropie...

Décision rendue

En fonction de la tâche, une fois arrivé dans la feuille de fin :

Classification classe majoritaire

Régression moyenne des valeurs cibles

Splits possibles

Splits possibles d'une feature donnée :

Catégorielle chaque catégorie vs le reste

Ordinale/Continue milieu de chaque valeur ou quantiles

Évaluation de la qualité d'un split

En fonction de la tâche :

Régression coût si on rendait la moyenne des instances comme résultat

$$Loss = \sum |\hat{y} - y| \approx variance$$

Classification Entropie de Shannon :

$$Loss = -\sum_{x \in X} P_x * \log_2(P_x)$$

$= 0 \Rightarrow$ il n'y a pas d'incertitude
maximale quand on a une distribution uniforme

Exemple — démarrage

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Première étape : création du nœud de départ

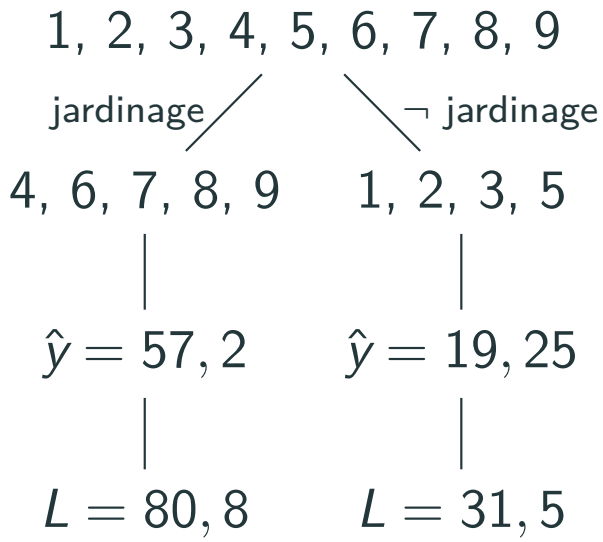
1, 2, 3, 4, 5, 6, 7, 8, 9

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur jardinage :



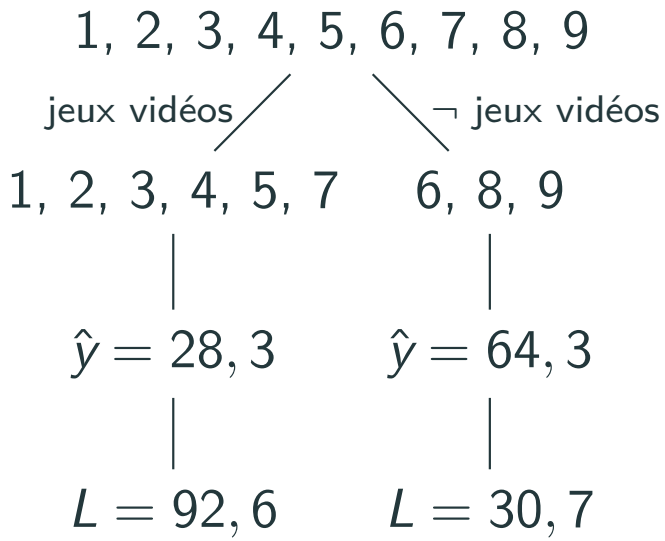
Loss totale : $122,3$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur jeux vidéos :



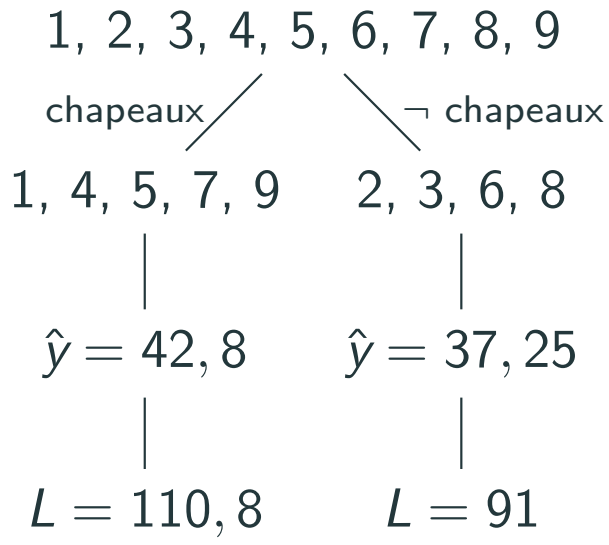
Loss totale : $123, 3$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur chapeaux :



Loss totale : $201,8$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

122,3 jardinage

123,3 jeux vidéos

201,8 chapeaux

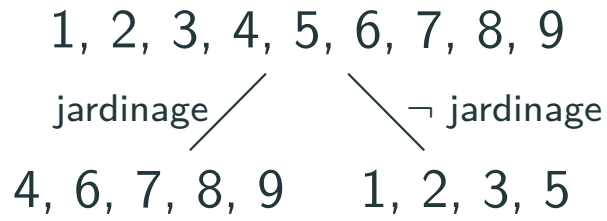
$\rightarrow$ On split donc sur jardinage

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Résultat après le premier split :



À vous de jouer !

Limiter l'overfit

Fait par :

- la profondeur maximum
- le nombre minimum d'instances dans chaque feuille
- une baisse d'entropie maximale à chaque split
- le nombre minimum d'instances pour split
- le pruning

Random Forest

Introduction

- les arbres de décision overfit facilement
- ils sont rapides à apprendre
- en combiner beaucoup est faisable et réduit la variance

$\rightarrow$ création d'une forêt (ensemble d'arbres) aléatoire

But

Produire des arbres décorrélés et moyenner leurs prédictions pour réduire la variance.

Outil 1 — bagging (row sampling)

Boostrap **agg**regat**ing** (Bagging) :

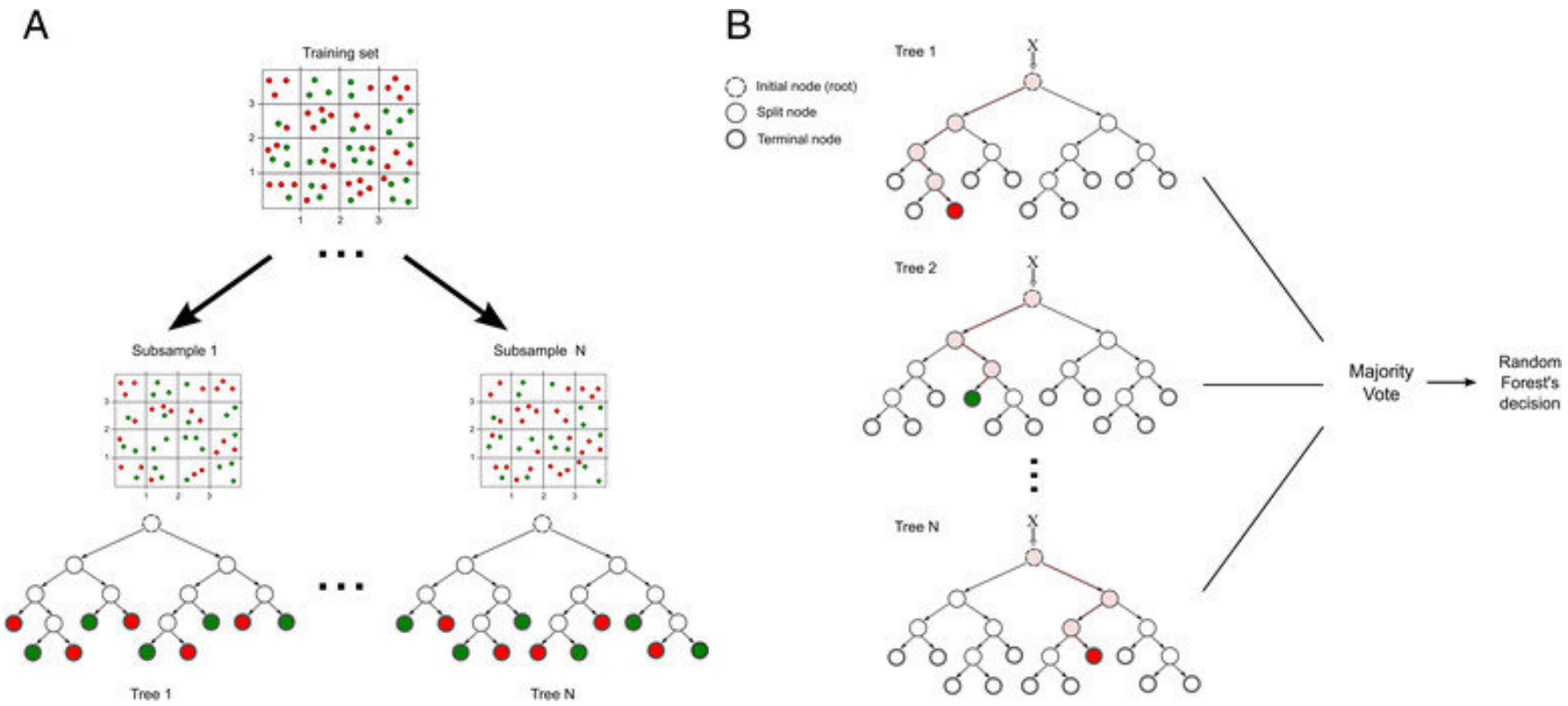
- tirer un échantillon du dataset avec replacement
- entraîner un arbre sur cet échantillon
- répéter B fois

Le bagging s'appelle aussi row sampling.

Outil 2 — random subspace method (column sampling)

- à chaque split, considérer seulement un sous-ensemble des features
- valeurs conseillées :
  - classification : $|\sqrt{m}|$ features par split
  - regréssion : $|\frac{m}{3}|$ features par split, 5 exemples par node minimum

Random Forest



Random Forest

- Pas de sur-apprentissage en augmentant le nombre d'arbres
- Une fois appris, le modèle est très rapide

Conclusion

Conclusion

- les arbres sont interprétables, rapides à entraîner, combinables.
- random forest combine des arbres faibles en un prédicteur versatile

TP 1

Exploration de données

TP 2 : Régression Linéaire

www.regression-linéaire.ipynb

TP 2 : Régression Logistique

www.regression-logistique.ipynb

TP 2 : SVM

www.svm.ipynb

TP 2 : Random Forest

www.random-forest-court.ipynb

TP 2 : Random Forest

www.random-forest-long.ipynb