LE MACHINE-LEARNING EN PRATIQUE

Vincent Guigue
vincent.guigue@agroparistech.fr

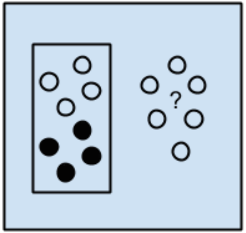
MIA PARIS-SACLAY
EKINOCS

AGROPARISTECH
Institut des Sciences et Industries du Vivant et de l'Environnement

Introduction

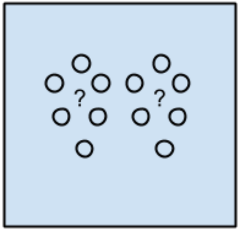
Différents cadres de machine learning

Supervisé



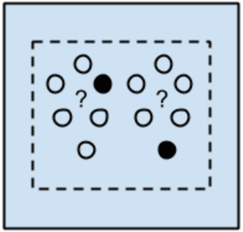
Supervised Learning Algorithms

Non-supervisé



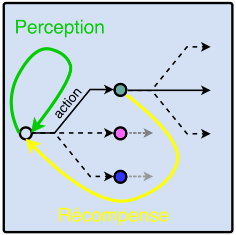
Unsupervised Learning Algorithms

Semi-supervisé



Semi-supervised Learning Algorithms

Renforcement



- Différents algorithmes...

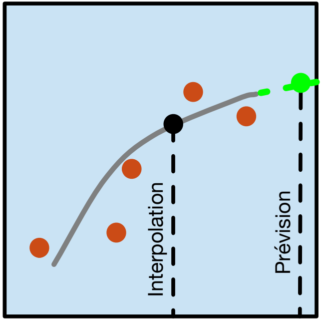
... et différentes évaluations

- Différentes **données**, différents **coûts**...

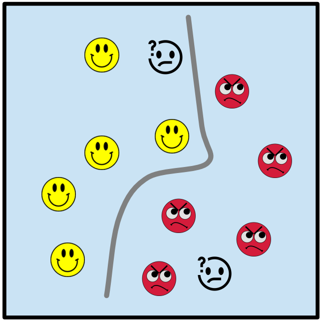
Et une nouvelle donne avec *Amazon Mechanical Turk*

Grande familles de problématiques supervisées

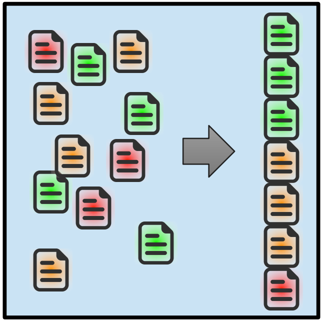
Régression



Classification

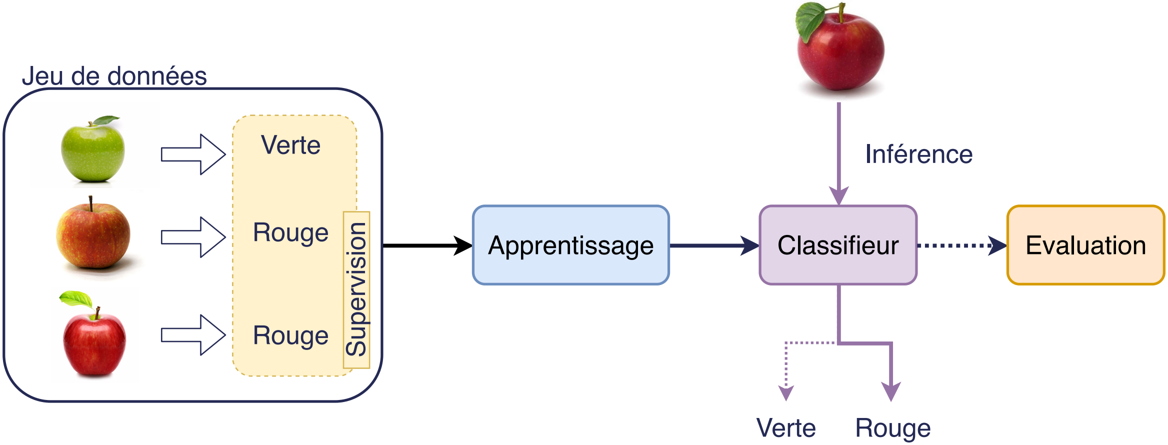


Ordonnancement



Chaine de traitement

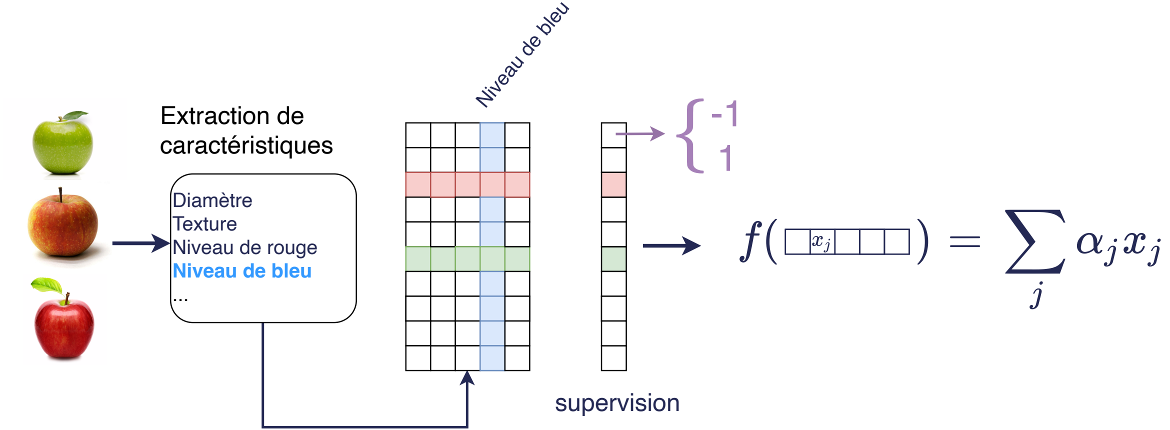
Identifier les entrées / sorties + évaluation



... En version abstraite

Chaine de traitement

En plus concret :



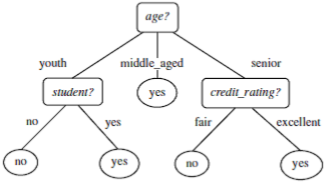
- Sélection des bonnes colonnes
- Ajout de colonnes intéressantes (calculs, sources de données externes, ...)

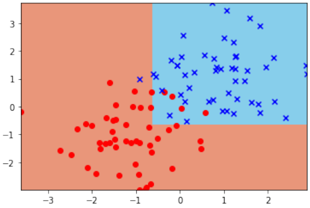
Classes de modèles

Modèles de ML : références historiques

Arbre de décision : entre IA symbolique & apprentissage statistique

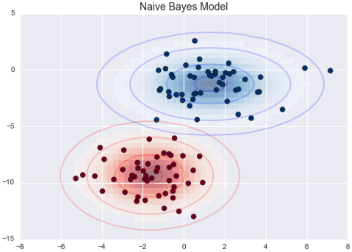
- Ensemble de règles
- Interprétable
  (selon la profondeur)
- Apprenable
  (sur critère entropique)

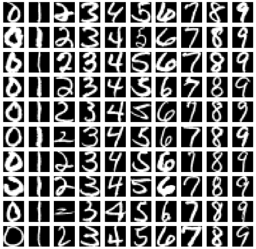




Modélisation bayesienne

- Lois de probabilité
- Max. de vraisemblance
- Naive Bayes
- A priori des experts

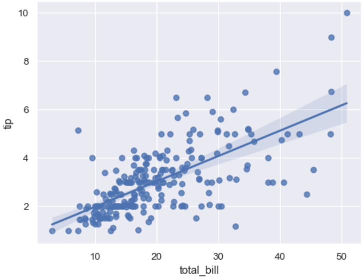


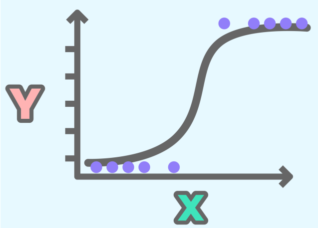


Modèles de ML : les bonnes affaires

- **Modèles linéaires : Moindre carrés (MSE), régression logistique, ...**

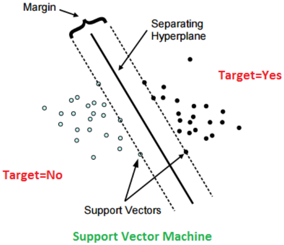
- Formulation simple & efficace
- Classif, régression
- Références très solides / modèle discriminant
- Descente de gradient

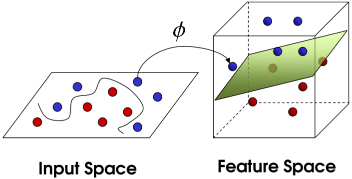




- **SVM, noyaux et méthodes discriminantes**

- Perceptron
- Régularisation
- SVM
- Projection non linéaire

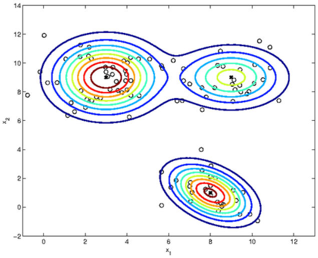


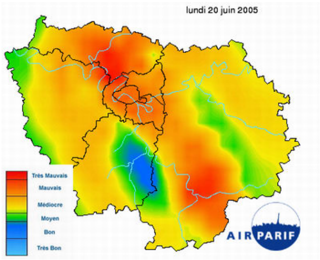


Modèles de ML : approches non-supervisées

Estimation de densité

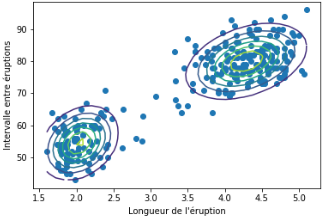
- Parzen
- Nadaraya-Watson
- Détour par les Knn
- EM

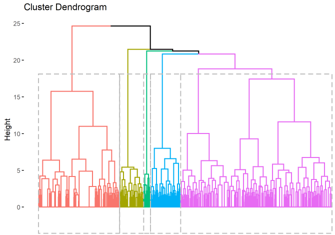




Clustering

- clustering hierarchique
- k-means / C-EM
- Clustering spectral
- A Priori

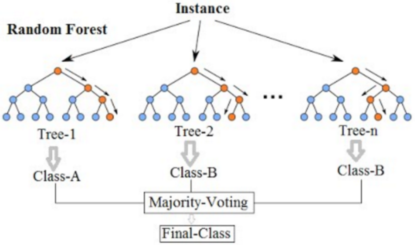




Modèles de ML : l'état de l'art

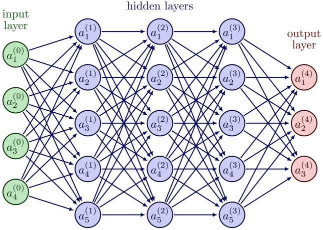
Approches ensemblistes

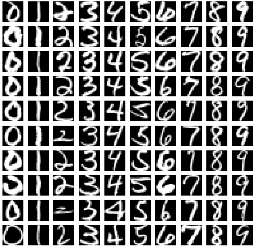
- Bagging
- Boosting
- Forêt, forêt aléatoire
- XGBoost



Réseaux de neurones (⇒ `pytorch`)

- Perceptron
- Réseaux de neurones
- Rétropropagation du gradient
- Différentes architecture





Focus sur les Arbres de décision

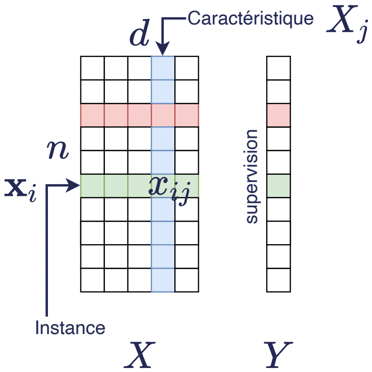
Notations usuelles en classification

On dispose :

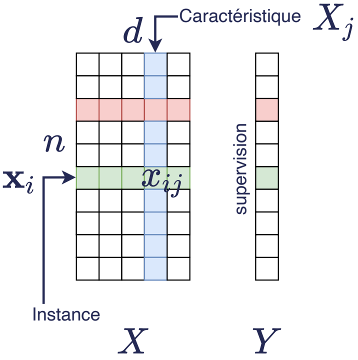
- Espace de représentation $\mathcal{X}$
  Une caractéristique/variable/attribut X_j
  peut être continue, ordinale ou discrète
  Souvent $\mathcal{X} = \mathbb{R}^d$
  d est la dimension de l'espace de représentation
- Ensemble d'exemples/instances
  $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n\}$, $\mathbf{x_i} \in \mathcal{X}$
  $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})$
- Supervision = étiquettes $Y = \{y_1, \ldots, y_i, \ldots, y_n\}$
  dans le cas binaire, $y_i \in \{0, 1\}$ ou $y_i \in \{-1, 1\}$

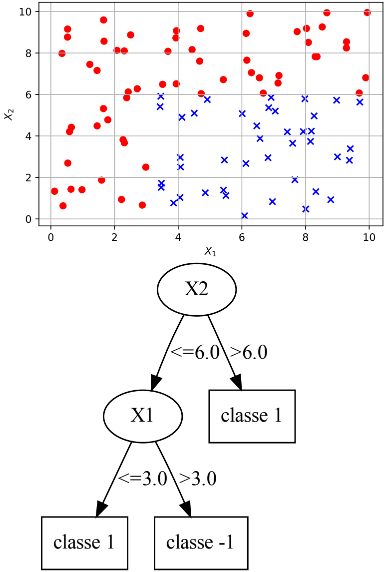
On veut :

Trouver une fonction $f : \mathcal{X} \to Y$ telle que la prédiction sur de futurs exemples soit la plus précise possible.



Principe de l'arbre de décision

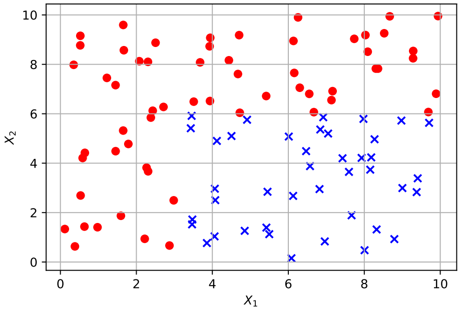




- Présenter une instance **x** à la racine
- Nœud = test d'une variable
- Branche = résultat du test
- Feuilles = étiquette y de l'instance

Algorithme général

classe ???

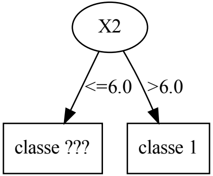


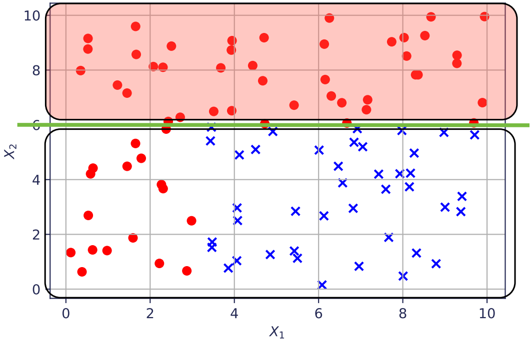
Algorithme glouton, top-down

Initialisation à la racine avec tous les exemples

- Si le nœud n'est pas pur, alors
  - Trouver X_j la **meilleure variable** pour ce nœud et le **test associé**
  - Pour chaque test, créer un fils au nœud courant
  - Faire *tomber* les exemples du nœud courant à leur fils correspondant
- sinon transformer le nœud en feuille.

Algorithme général



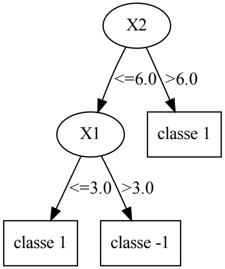


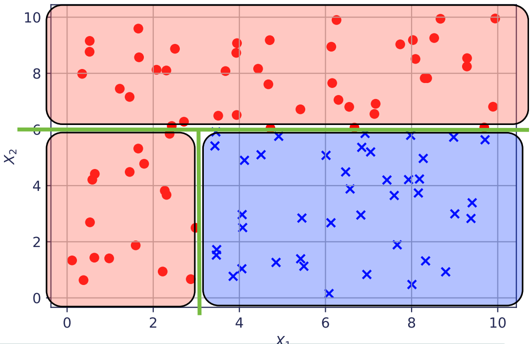
Algorithme glouton, top-down

Initialisation à la racine avec tous les exemples

- Si le nœud n'est pas pur, alors
  - Trouver X_j la **meilleure variable** pour ce nœud et le **test associé**
  - Pour chaque test, créer un fils au nœud courant
  - Faire *tomber* les exemples du nœud courant à leur fils correspondant
- sinon transformer le nœud en feuille.

Algorithme général



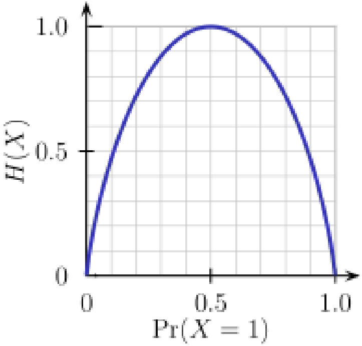


Algorithme glouton, top-down

Initialisation à la racine avec tous les exemples

- Si le nœud n'est pas pur, alors
  - Trouver X_j la **meilleure variable** pour ce nœud et le **test associé**
  - Pour chaque test, créer un fils au nœud courant
  - Faire *tomber* les exemples du nœud courant à leur fils correspondant
- sinon transformer le nœud en feuille.

Sélectionner la meilleure variable



Entropie d'une variable aléatoire

Soit X une variable aléatoire pouvant prendre n valeurs x_i :

$$H(X) = -\sum_{i=1}^{n} P(X = x_i) log(P(X = x_i))$$

Entropie $\nearrow$ = désordre $\nearrow$

Entropie nulle $\rightarrow$ pas d'aléa

$\rightarrow$ classification parfaite.

Sélectionner la meilleure variable

Entropie d'un échantillon : cas binaire

- X un ensemble de données, Y leurs étiquettes (positif/négatif)
- p_+ la proportion d'exemples positifs
- p_- la proportion d'exemples négatifs
- $H(Y) = -p_+ log(p_+) - p_- log(p_-)$

Entropie conditionnelle

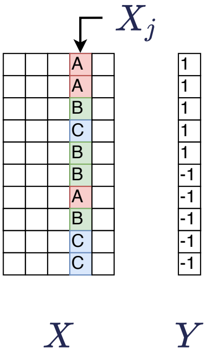
- Entropie conditionnelle : $H(Y|X) = \sum_i P(X = x_i) H(Y|X = x_i)$
- Un test T sur une variable $\Rightarrow$ deux partitions d'exemples de X : $X^{(1)}$ qui vérifie le test et $X^{(2)}$ qui ne vérifie pas le test (resp. $Y^{(1)}$ et $Y^{(2)}$). L'entropie conditionnelle au test T est :

$$H(Y|T) = \frac{|X^{(1)}|}{|X|} H(Y^{(1)}) + \frac{|X^{(2)}|}{|X|} H(Y^{(2)})$$

$\Rightarrow$ Gain d'information : $I(T, Y) = H(Y) - H(Y|T)$ à maximiser

$\Leftrightarrow$ minimiser $H(Y|T)$

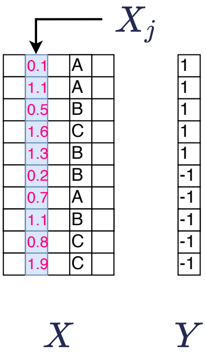
Cas discret / Cas continu



Cas discret

$X_j \in \{A, B, C\} \Rightarrow$ Ensemble d'exemples divisé en 3 $\Rightarrow$ Calcul aisé de l'entropie :

$$H(Y|X_j) = \frac{|A|}{|X|}H(Y^{(A)}) + \frac{|B|}{|X|}H(Y^{(B)}) + \frac{|C|}{|X|}H(Y^{(C)})$$



Cas continu

$X_j \in [0, 2] \Rightarrow$ il faut tester toutes les valeurs de coupure !

1. Ordonnancement des valeurs de X_j
2. Calcul de la valeur de $H(Y|X_j)$ pour tous les tests
3. Conservation de la meilleure valeur

Cas discret / Cas continu

X_j

			A	
			A	
			B	
			C	
			B	
			B	
			A	
			B	
			C	
			C	

1
1
1
1
1
-1
-1
-1
-1
-1

X Y

X_j

	0.1		A	
	1.1		A	
	0.5		B	
	1.6		C	
	1.3		B	
	0.2		B	
	0.7		A	
	1.1		B	
	0.8		C	
	1.9		C	

1
1
1
1
1
-1
-1
-1
-1
-1

X Y

Cas discret

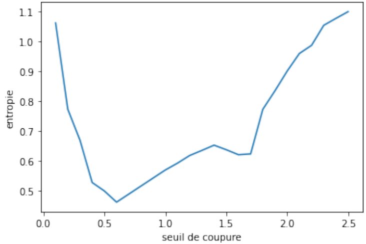
$X_j \in \{A, B, C\} \Rightarrow$ Ensemble d'exemples divisé en 3 $\Rightarrow$ Calcul aisé de l'entropie :

$$H(Y|X_j) = \frac{|A|}{|X|}H(Y^{(A)}) + \frac{|B|}{|X|}H(Y^{(B)}) + \frac{|C|}{|X|}H(Y^{(C)})$$

Cas continu

Courbe type :

Entropie *vs* coupure



Focus sur les Support Vector Machine

Focus sur les SVM

$X = \{\mathbf{x}_1, \dots, \mathbf{x}_i, \dots, \mathbf{x}_n\}$, $\mathbf{x_i} \in \mathbb{R}^d$, $Y = \{y_1, \dots, y_i, \dots, y_n\}$, $y_i \in \{-1, 1\}$

- **Coût** (discriminant) :

$$\mathcal{L} = \sum_{i=1}^{n} (1 - y_i f(\mathbf{x}_i))_+, \qquad (a)_+ = \max(a, 0) = \text{ Partie positive}$$

  - Quelle différence avec les moindes carrés ?
- Forme de la décision (duale/par rapport aux points d'apprentissage) :

$$f(\mathbf{x}_i) = \sum_{j=1}^{n_{app}} w_j \cdot k(\mathbf{x}_i, \mathbf{x}_j)$$

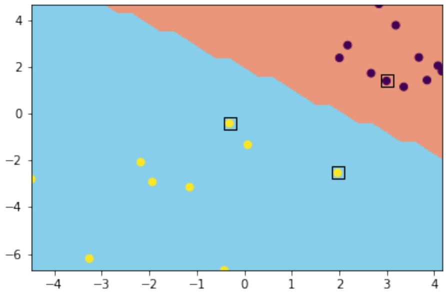
  - k : kernel/noyau [linéaire = produit scalaire, polynomial, gaussien, ...]
- **Régularisation :**

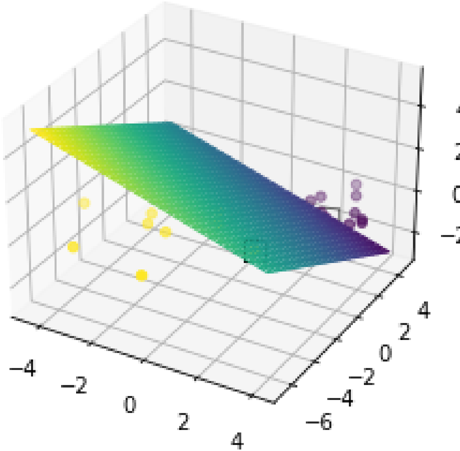
$$\mathcal{L} = \sum_{i=1}^{n} (1 - y_i f(\mathbf{x}_i))_+ + C\|\mathbf{w}\|^2$$

  - Hyper-paramètre C
  - Réflexion sur les cas extrêmes

Notion de vecteur support de la décision

Cas linéaire :



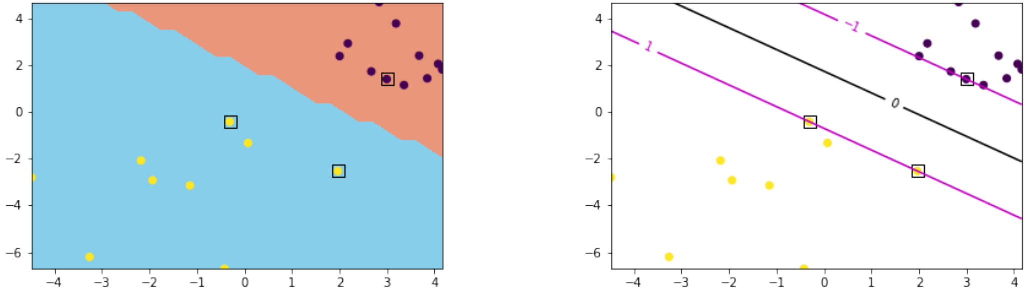


La décision pour l'ensemble de l'espace ne repose que sur 2 vecteurs supports

$\Rightarrow$ 2 points d'apprentissage ont été sélectionnés

Notion de marge

Cas linéaire :

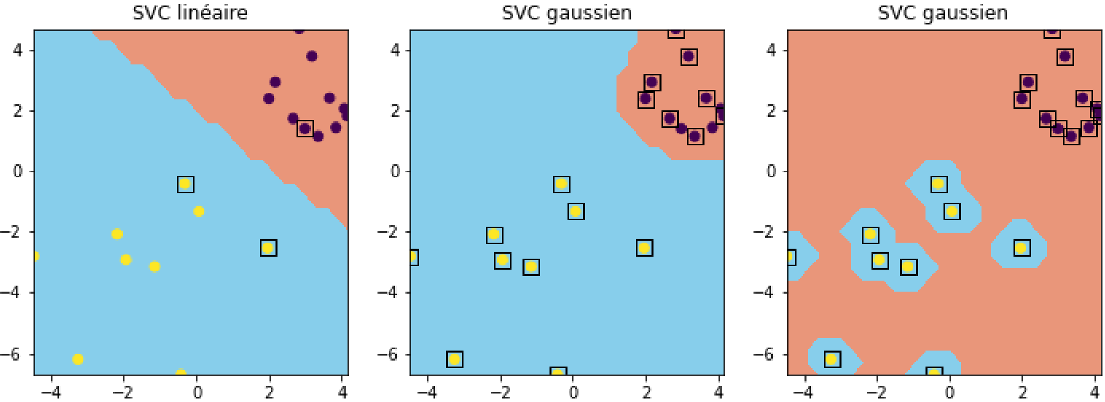


$$\mathcal{L} = \sum_{i=1}^{n} (1 - y_i f(\mathbf{x}_i))_+ + C\|\mathbf{w}\|^2$$

- Les vecteurs supports sont sur et dans la **marge**
  - $=$ dans la zone de coût non nul
- Distinguer les cas **séparable** et **non-séparable**

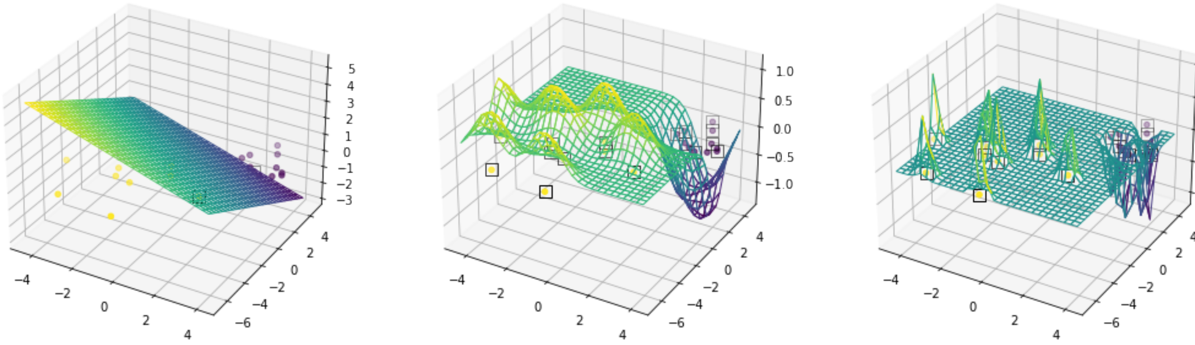
Lien avec les mixtures de gaussiennes

En prenant : $f(\mathbf{x}_i) = \sum_{j=1}^{n_{app}} w_j \cdot k(\mathbf{x}_i, \mathbf{x}_j)$, $\qquad k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$

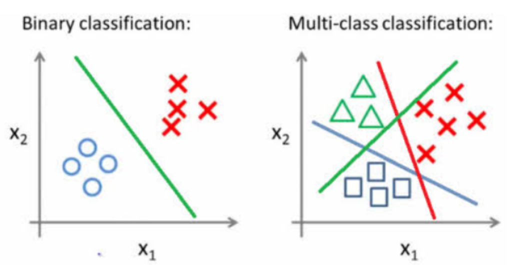


Lien avec les mixtures de gaussiennes

En prenant : $f(\mathbf{x}_i) = \sum_{j=1}^{n_{app}} w_j \cdot k(\mathbf{x}_i, \mathbf{x}_j)$, $\qquad k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$



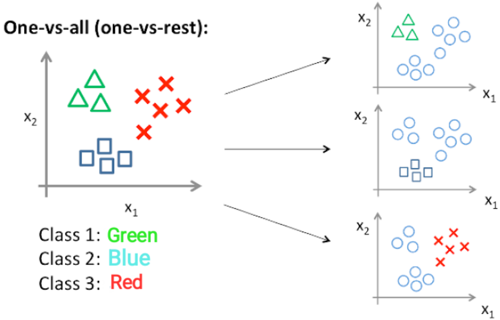
Du cas binaire au multi-classes



Nombreux classifieurs binaires (dont les SVM)

Nombreux problèmes multi-classes

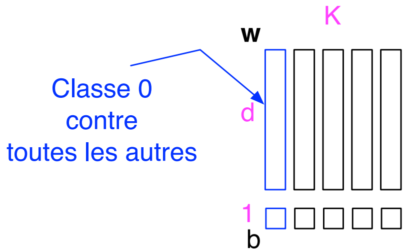
Comment faire ?



Un-contre-tous (one-against-all)

Multi-classe (2)

un contre tous (*one against all*) : K classes
$\Rightarrow$ K classifieurs appris séparement sur
toutes les données



- $f(\mathbf{x}) \Rightarrow f_k(\mathbf{x})$ et critère de décision :

$$k^\star = \arg\max_k f_k(\mathbf{x})$$

Quelle classe veut **le plus** l'échantillon $\mathbf{x}$?

- Critères de rejet :
  - pas de $f_k(\mathbf{x}) > 0.5$
  - plusieurs $f_k(\mathbf{x}) > 0.5$

SVM, quelques conclusions

- La méthode de référence des années 90+2000
- Plein de noyaux/kernel pour différents types de données
  - Graphes, images, ...
- Initialement une méthode de classification (SVM, SVC). Des extensions pour la régression (SVR).
- Problème majeur de passage à l'échelle
  - Complexité en $\mathcal{O}(n^2)$ au mieux... Souvent $\mathcal{O}(n^3)$ ou $\mathcal{O}(n^4)$

Approches ensemblistes

- Bagging
- Boosting

- Random Forest
- XGBoost

Conclusion

Conclusion

scikit-learn est un outil très puissant qui

- Propose des modèles sur l'étagère (supervisé, non-supervisé, estimation de densité, ...)
- Propose des métriques, des pré-traitements, des chaines...
- Parallélise les calculs (cf `njob`)
- Vous permet d'insérer vos outils dans ce cadre

Allons voir ça de plus près