Les SVMs contre le Deep Learning

Clément Dechesne

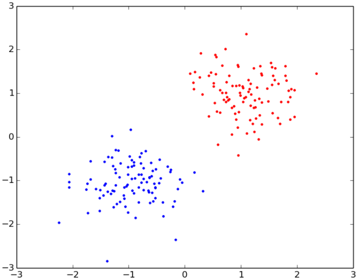
Atelier machine learning

12 mars 2019

Plan

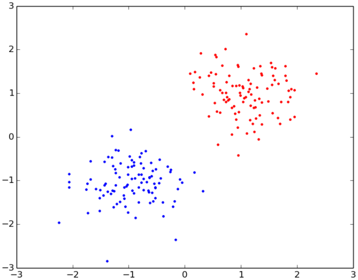
SVM

Objectif : Trouver une séparation linéaire entre deux ensembles



SVM

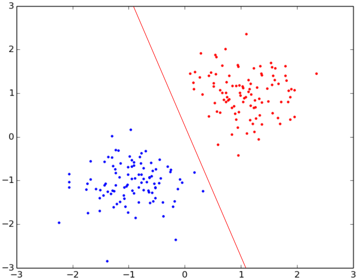
Objectif : Trouver une séparation linéaire entre deux ensembles



$$
\begin{aligned}
D(x) &= \mathrm{sign}(f(\mathbf{x})) \\
&= \mathrm{sign}(\mathbf{v}^{\mathsf{T}}\mathbf{x} + a)
\end{aligned}
$$

SVM

Objectif : Trouver une séparation linéaire entre deux ensembles

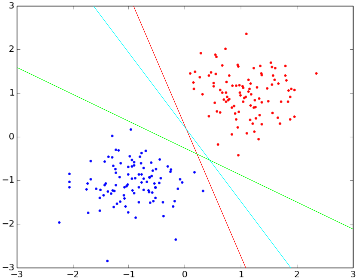


$$\begin{aligned} D(x) &= \text{sign}(f(\mathbf{x})) \\ &= \text{sign}(\mathbf{v}^\top \mathbf{x} + a) \end{aligned}$$

Une ligne de décision : $\mathbf{v}^\top \mathbf{x} + a = 0$

SVM

Objectif : Trouver une séparation linéaire entre deux ensembles



$$D(x) = \text{sign}(f(\mathbf{x})) = \text{sign}(\mathbf{v}^\top \mathbf{x} + a)$$

Une ligne de décision : $\mathbf{v}^\top \mathbf{x} + a = 0$

Mais il en existe beaucoup d'autres

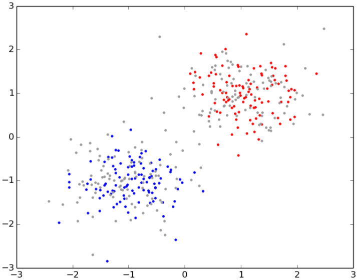
Le problème est **mal posé**

Comment trouver la meilleure solution ?

SVM

Le problème est un peu différent

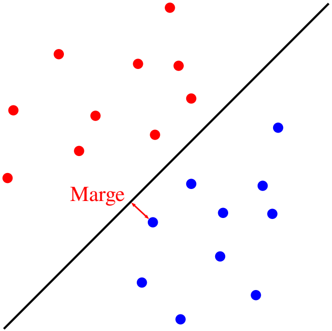
Soit un ensemble d'apprentissage : $\{\mathbf{x}_i, y_i\}_{i \in [1,n]}$



On veut être capable de classifier des observations nouvelles en minimisant les **erreurs**.

Le problème est un peu différent

Soit un ensemble d'apprentissage : $\{\mathbf{x}_i, y_i\}_{i \in [1,n]}$



On veut être capable de classifier des observations nouvelles en minimisant les **erreurs**.

$D(x) = \text{sign}(\mathbf{v}^T \mathbf{x} + a)$

Comment trouver la meilleure solution ?

$\Rightarrow$ **maximiser la marge**

Vapnik (1982)

La marge d'un SVM

SVM

Marge

SVM linéaire

Erreurs potentielles

Cross-validation

SVM non linéaire

SVM vs Deep Learning

Maximiser la confiance = Maximiser la marge

La frontière de décision qui maximise la marge est :

$\Delta(\mathbf{v}, a) = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{v}^T \mathbf{x} + a = 0\}$

$$\max_{\mathbf{v},a} \underbrace{\min_{i \in [i,n]} \text{dist}(\mathbf{x}_i, \Delta(\mathbf{v}, a))}_{\text{marge}:m}$$

La marge d'un SVM

Maximiser la confiance = Maximiser la marge

La frontière de décision qui maximise la marge est :

$\Delta(\mathbf{v}, a) = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{v}^T\mathbf{x} + a = 0\}$

$$\max_{\mathbf{v},a} \underbrace{\min_{i \in [i,n]} \mathrm{dist}(\mathbf{x}_i, \Delta(\mathbf{v}, a))}_{\text{marge:m}}$$

Maximiser la marge

$$\begin{cases} \max_{\mathbf{v},a} m \\ \text{avec } \min_{i \in [i,n]} \frac{|\mathbf{v}^T\mathbf{x}+a|}{||\mathbf{v}||} \geqslant m \end{cases}$$

La marge d'un SVM

SVM

Marge

SVM linéaire

Erreurs potentielles

Cross-validation

SVM non linéaire

SVM vs Deep Learning

Maximiser la confiance = Maximiser la marge

La frontière de décision qui maximise la marge est :

$\Delta(\mathbf{v}, a) = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{v}^T\mathbf{x} + a = 0\}$

$$\max_{\mathbf{v},a} \underbrace{\min_{i \in [i,n]} \mathrm{dist}(\mathbf{x}_i, \Delta(\mathbf{v}, a))}_{\text{marge}:m}$$

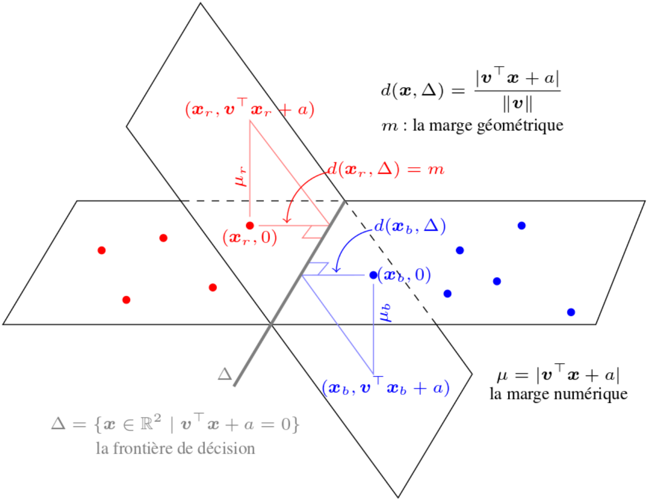
Maximiser la marge

$$\begin{cases} \max\limits_{\mathbf{v},a} m \\ \text{avec } \min\limits_{i \in [i,n]} \frac{|\mathbf{v}^T\mathbf{x}+a|}{||\mathbf{v}||} \geqslant m \end{cases}$$

Problème encore mal posé :

si $(\mathbf{v}, a)$ est une solution, alors $\forall k > 0 (k\mathbf{v}, ka)$ est aussi solution

La marge d'un classifieur : marge numérique et marge géométrique



La marge d'un SVM

Maximiser la marge :

$$\begin{cases} \max\limits_{\mathbf{v},a} m \\ \text{avec } \min\limits_{i\in[i,n]} |\mathbf{v}^T\mathbf{x}_i + a| \geqslant m \\ \|\mathbf{v}\|^2 = 1 \end{cases}$$

La marge d'un SVM

Maximiser la marge :

$$\begin{cases} \max\limits_{\mathbf{v},a} m \\ \text{avec } \min\limits_{i \in [i,n]} |\mathbf{v}^T \mathbf{x}_i + a| \geqslant m \\ {\color{red}||\mathbf{v}||^2 = 1} \end{cases}$$

Si le minimum est supérieur à la marge, alors tous les $|\mathbf{v}^T \mathbf{x}_i + a|$ sont supérieur à la marge et $y_i \in \{-1, 1\}$

$$\begin{cases} \max\limits_{\mathbf{v},a} m \\ \text{avec } y_i(\mathbf{v}^T \mathbf{x}_i + a) \geqslant m \quad \forall i \in [1, n] \\ ||\mathbf{v}||^2 = 1 \end{cases}$$

La marge d'un SVM

$$\begin{cases} \max\limits_{\mathbf{v},a} m \\ \text{avec } y_i(\mathbf{v}^T\mathbf{x}_i + a) \geqslant m \quad \forall i \in [1, n] \\ ||\mathbf{v}||^2 = 1 \end{cases}$$

On effectue le changement de variable $\mathbf{w} = \frac{\mathbf{v}}{m}$ et $b = \frac{a}{m}$
$\Longrightarrow ||\mathbf{w}|| = \frac{||\mathbf{v}||}{m}$

La marge d'un SVM

SVM

Marge

SVM linéaire

Erreurs potentielles

Cross-validation

SVM non linéaire

SVM vs Deep Learning

$$\begin{cases} \max_{\mathbf{v},a} m \\ \text{avec } y_i(\mathbf{v}^T\mathbf{x}_i + a) \geqslant m \quad \forall i \in [1, n] \\ ||\mathbf{v}||^2 = 1 \end{cases}$$

On effectue le changement de variable $\mathbf{w} = \frac{\mathbf{v}}{m}$ et $b = \frac{a}{m}$

$\implies ||\mathbf{w}|| = \frac{||\mathbf{v}||}{m}$

$$\begin{cases} \max_{\mathbf{w},b} m \\ \text{avec } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 \quad \forall i \in [1, n] \\ m = \frac{1}{||\mathbf{w}||} \end{cases}$$

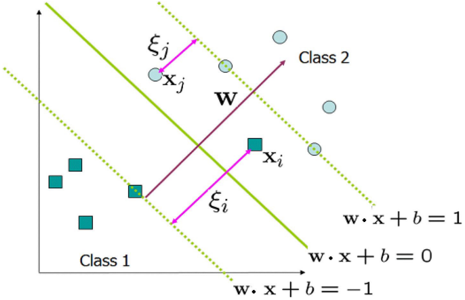
$$\begin{cases} \min_{\mathbf{w},b} ||w||^2 \\ \text{avec } \underset{\forall i \in [1,n]}{y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1} \end{cases}$$

Définition SVM linéaire

Définition : Soit $\{(\mathbf{x}_i, y_i); i \in [1, n]\}$ un ensemble de vecteur formes étiquetés avec $\mathbf{x}_i \in \mathbb{R}^d$ et $y_i \in \{-1, 1\}$. Un séparateur à vaste marge linéaire (SVM) est un discriminateur linéaire de la forme : $D(x) = sign(\mathbf{w}^T\mathbf{x} + b)$ où $\mathbf{w} \in \mathbb{R}^d$ et $b \in \mathbb{R}$ sont donnés par la résolution du problème suivant :

$$\begin{cases} \min\limits_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 \\ \text{avec } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 \quad \forall i \in [1, n] \end{cases}$$

Erreurs potentielles



Erreurs potentielles

SVM

Marge

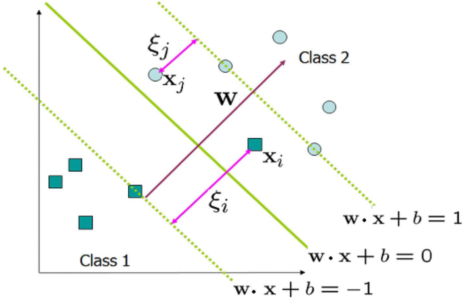
SVM linéaire

Erreurs potentielles

Cross-validation

SVM non linéaire

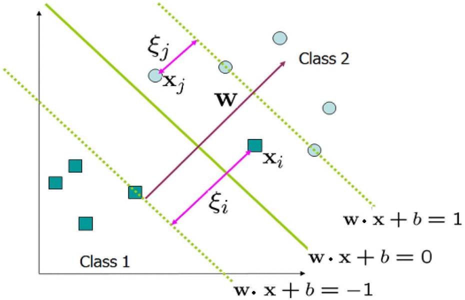
SVM vs Deep Learning



Idée : Modéliser les erreurs potentielles

$$(\mathbf{x}_i, y_i) \begin{cases} \text{pas d'erreur : } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 \Longrightarrow \xi_i = 0 \\ \text{erreur : } \xi_i = 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) > 0 \end{cases}$$

Erreurs potentielles



Nouveau problème à résoudre :

$$\begin{cases} \min\limits_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{p}\sum_{i=1}^{n} \xi_i^p \quad p = 1 \text{ ou } 2 \\ \text{avec } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 - \xi_i \quad \forall i \in [1, n] \\ \qquad \xi_i \geqslant 0 \end{cases}$$

Erreurs potentielles

Le terme d'équilibrage $C > 0$ permet de fixer le coût d'une erreur :

Si C **est fort**, on tolère peu d'erreurs, la marge est plus faible

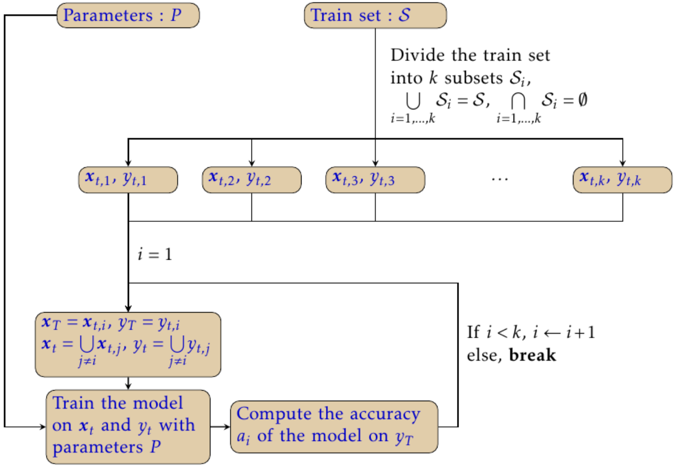
$\Longrightarrow$ **moins bonne généralisation**

Si C **est faible**, on tolère beaucoup d'erreurs, la marge est plus grande

$\Longrightarrow$ **meilleure généralisation**

Trouver un **bon** C est crucial.
Une méthode possible : la **cross-validation**

Cross-validation



The accuracy for parameters P is $A = \frac{1}{k}\sum_{i=1}^{k} a_i$. The optimal set of parameters are chosen with regard to the greatest accuracy A.

Principe des noyaux

Dans la plupart des cas, la frontière optimale n'est pas linéaire. Dans le cadre des SVM, la prise en compte de non linéarités dans le modèle s'effectue par l'introduction de noyaux non linéaires. Ce noyau est une fonction K qui associe à tout couple d'observations $(\mathbf{x}_i, \mathbf{x}_j)$ une mesure de leur « influence réciproque » calculée à travers leur corrélation ou leur distance ; $K(\mathbf{x}_i, \mathbf{x}_j) = (h(\mathbf{x}_i))^T h(\mathbf{x}_j)$.

Noyaux non linéaires

Des exemples typiques de noyaux sont :
le noyaux polynomial : $K(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^d$,
le noyau gaussien : $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}$.

Dans l'espace des attributs transformés, $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i h(\mathbf{x}_i)$.

Donc $f(\mathbf{x}) = \mathbf{w}^T h(\mathbf{x}) + b = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$

Plan

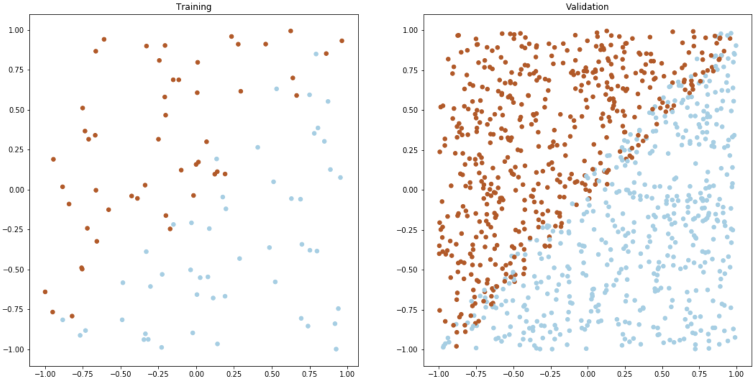
7. SVM vs Deep Learning
   - Datasets
   - Réseau
   - Un peu de code...
   - Résultats

Dataset

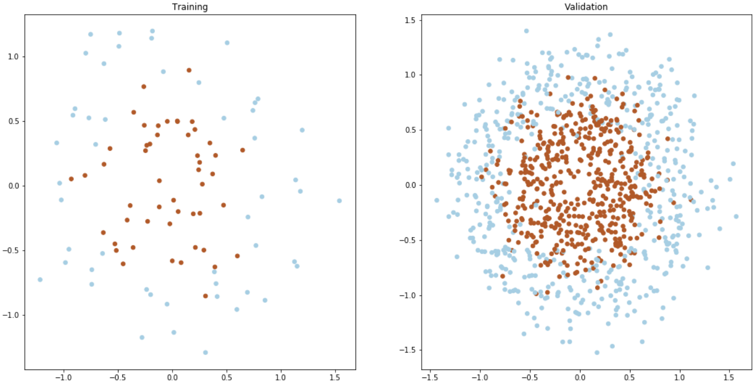
Utilisation de données synthétiques a 2 attributs (nuage de points) :

+ Contrôle du nombre d'échantillons
+ Interprétation visuelle
– Données "simple"
– Intérêt limité

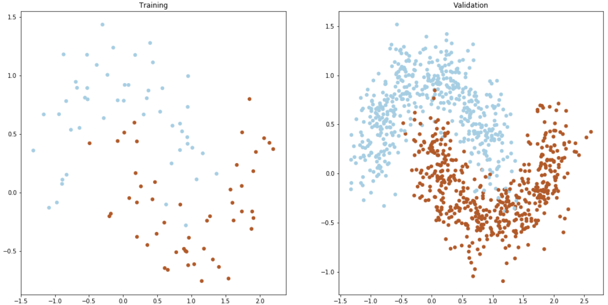
Dataset - linéaire



Dataset - circulaire



Dataset - lunes



Réseau employé

Réseau simple (tâche peu complexe)

- 4 couches denses ayant le même nombre d'unités cachées
- activation ReLu

Code

Les SVM sous python :

Module SVC de sklearn :	`from sklearn.svm import SVC`
Créer un objet SVM :	`clf = SVC(kernel="linear", C=c)`
	`clf = SVC(kernel="rbf", C=c, gamma=g)`
Entraîner un SVM :	`clf.fit(X, y)`
Score du SVM :	`clf.score(X, y)`

Code

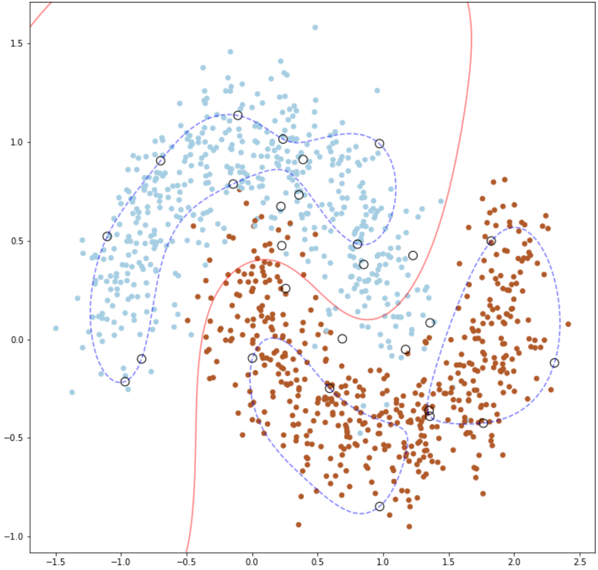
Les SVM sous python :

Module SVC de sklearn : `from sklearn.svm import SVC`
Créer un objet SVM :
```
clf = SVC(kernel="linear", C=c)
clf = SVC(kernel="rbf", C=c, gamma=g)
```
Entraîner un SVM : `clf.fit(X, y)`
Score du SVM : `clf.score(X, y)`

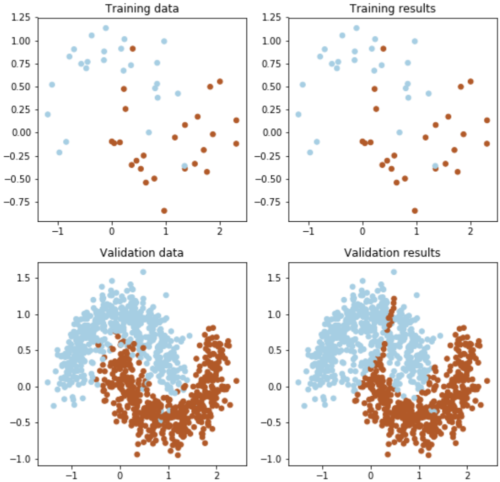
La cross validation :

Module KFold de sklearn : `from sklearn.model_selection import KFold`
Créer un ensemble : `kf = KFold(n_splits=N)`
Parcourir l'ensemble : `for train_index, test_index in kf.split(X):`

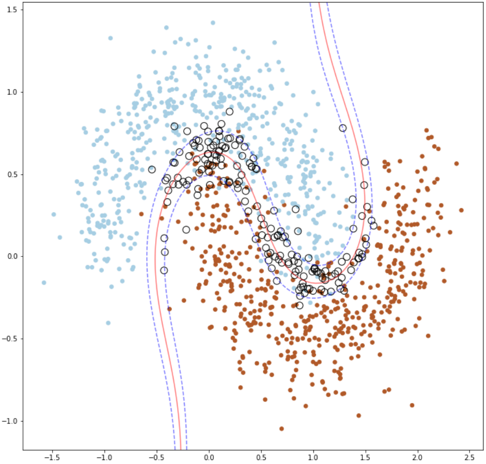
Résultats - SVM - 50 échantillons



Résultats - DL - 50 échantillons



Résultats - SVM - 2000 échantillons



Résultats - DL - 2000 échantillons