Classification — la fin

Pipeline d'apprentissage et SVM

Guillaume Wisniewski
guillaume.wisniewski@limsi.fr

Université Paris Sud — LIMSI

Mars 2016

One theorem to rule them all...

- Tout l'apprentissage repose sur un théorème :

$$\epsilon_{\text{test}} \leq \hat{\epsilon}_{\text{train}} + \sqrt{\frac{\text{complexity}}{n}} \quad (1)$$

- Lien entre l'erreur estimée sur un corpus de n exemples et l'erreur de généralisation (sur l'ensemble des données / non estimée)
- dépend de la complexité de la classe de fonctions considérée
- Avec des hypothèses très précises

Interprétation

- il n'y pas de fonctions qui permet d'obtenir une meilleure erreur en généarlisation que celle qui minimise l'erreur d'apprentissage...

Interprétation

- il n'y pas de fonctions qui permet d'obtenir une meilleure erreur en généarlisation que celle qui minimise l'erreur d'apprentissage...
- ...avec une forte probabilité...

Interprétation

- il n'y pas de fonctions qui permet d'obtenir une meilleure erreur en généarlisation que celle qui minimise l'erreur d'apprentissage...
- ...avec une forte probabilité...
- ...à l'intérieur d'une classe de fonctions données...

Interprétation

- il n'y pas de fonctions qui permet d'obtenir une meilleure erreur en généarlisation que celle qui minimise l'erreur d'apprentissage...
- ...avec une forte probabilité...
- ...à l'intérieur d'une classe de fonctions données...
- ...si les exemples d'apprentissage sont représentatifs.

1er message : Small is Beautiful

Apprendre, c'est minimiser

Parce que le monde est bien fait...

- classification binaire avec $\ell^{0/1}$
- minimiser l'erreur sur un ensemble d'apprentissage d'apprentissage $\Rightarrow$ problème NP-difficile
- optimisation directe impossible (ou pas)

http://jmlr.org/proceedings/papers/v28/nguyen13a.pdf

Première partie I

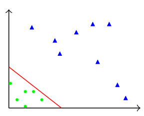
Retour sur le perceptron

Perceptron

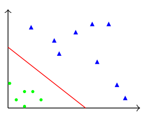
- Méthode heuristique d'optimisation de la fonction de $\ell^{0/1}$
- aucune garantie sur la convergence (sauf si les données sont linéairement séparable)
- aucune garantie sur la qualité de la solution trouvée

Les limites du perceptron

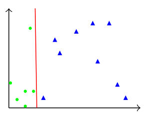
Quelle garantie en généralisation ?



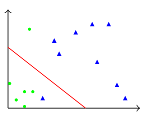
mauvaise solution ?



meilleure solution ?



solution trop spécifique



solution préférable

Grâces soient rendues aux Mathématiques

- aucun lien théorique entre erreur d'apprentissage et erreur de généralisation
- solution pragmatique : estimer expérimentalement l'erreur en généralisation
- en pratique, 3 ensembles indépendants :
  1. corpus d'apprentissage : permet de déterminer les paramètres du classifieur
  2. corpus de validation / développement : permet de monitorer l'erreur en généralisation :
     - ★ régulièrement, on calcule l'erreur sur l'ensemble de validation
     - ★ on conserve la valeur des paramètres qui minimise l'erreur de validation
  3. corpus de test : permet d'estimer l'erreur de généralisation (1 seule fois)

Pourquoi ça marche ?

- estimation non biaisée de l'erreur de généralisation (cf. théorème central limite)
- l'erreur est une moyenne ⇒ peut-être estimée à partir d'un échantillon suffisament grand
- mais nécessite plus de données annotées et un surcoût en temps de calcul

Problème de stabilité du perceptron

Le problème

- 10 000 exemples, après le 100e le perceptron est tellement bon qu'il ne fait pas d'erreurs sur les 9 899 exemples suivants...
- erreur sur le 10 000e exemple ⇒ la mise à jour va détruire le vecteur de poids qui avait parfaitement marcher sur 99.99% des exemples

La solution

- `random.shuffle`
- perceptron moyenné

Perceptron Moyenné

- Attention : pendant l'apprentissage on utilise la version « non moyennée » du perceptron

Conclusions (locales)

- peu de garanties théoriques...
- ...mais on sait faire marcher en pratique

Deuxième partie II

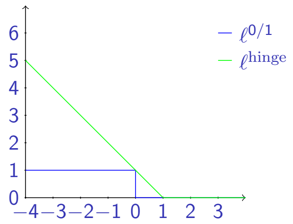
Support Vector Machine

Notre problème

$$\min_{\mathbf{w}} \sum \mathbb{1}\left\{f(\mathbf{x}_i;\mathbf{w}) \neq y_i\right\} \quad (2)$$

- problème NP-difficile

2E MESSAGE : LA NATURE DE L'APPRENTISSAGE STATISTIQUE



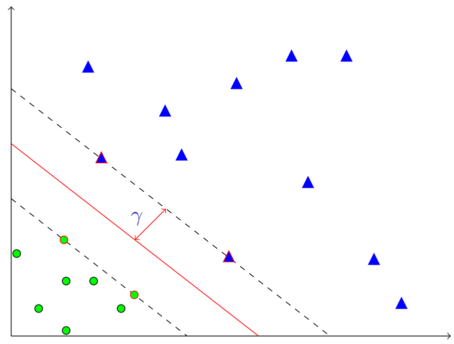
- considérer la hinge loss au lieu de $\ell^{0/1}$:

$$\ell^{\text{hinge}} = \max(0, 1 - t \cdot y) \tag{3}$$

où : t est l'étiquette de référence et y l'étiquette prédite ($y = \mathbf{x} \cdot \mathbf{w}$)

- minimiser la hinge revient à minimiser le $\ell^{0/1}$.

INTUITION DU SVM



$\Rightarrow$ maximiser la marge γ

MAXIMISER LA MARGE

POURQUOI LA MARGE ?

- trouver l'hyperplan séparateur le « plus loin » des données
- évite que les données inconnues ne soient trop proches de la frontière de décision

FORMALISATION : MARGE

- hyperplan séparateur : $\mathbf{w} \cdot \mathbf{x} = 0$
- hyperplan support : hyperplans parallèles à $\mathcal{H}$ passant par un exemple positif/négatifs et le plus proche de $\mathcal{H}$:

$$\mathbf{w} \cdot \mathbf{x} = \pm b \Leftrightarrow \mathbf{w} \cdot \mathbf{x} = \pm 1$$

- marge : distance entre $\mathcal{H}$ et l'hyperplan support :

$$\gamma = \frac{2}{||\mathbf{w}||}$$

avec de simples considérations géométriques

CRITÈRE D'APPRENTISSAGE

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations

CRITÈRE D'APPRENTISSAGE

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :

CRITÈRE D'APPRENTISSAGE

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale

Critère d'apprentissage

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale
   - tous les exemples sont du bon côté de leur hyperplan support

Critère d'apprentissage

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale
   - tous les exemples sont du bon côté de leur hyperplan support
   - pour les exemples positifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$

Critère d'apprentissage

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale
   - tous les exemples sont du bon côté de leur hyperplan support
   - pour les exemples positifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$
   - pour les exemples négatifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \leq -1$

Critère d'apprentissage

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale
   - tous les exemples sont du bon côté de leur hyperplan support
   - pour les exemples positifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$
   - pour les exemples négatifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \leq -1$
   - de manière compacte : $y^{(i)} \cdot \left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$

Critère d'apprentissage

1. maximiser la marge : $\max \frac{2}{||\mathbf{w}||} = \min ||\mathbf{w}||^2$ le carré permet juste de simplifier les équations
2. sous les contraintes :
   - s'il n'y a pas de contraintes : solution triviale
   - tous les exemples sont du bon côté de leur hyperplan support
   - pour les exemples positifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$
   - pour les exemples négatifs : $\left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \leq -1$
   - de manière compacte : $y^{(i)} \cdot \left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$
   - intuitivement : garanti que l'exemple est bien classé avec une marge de 1 cf. perceptron

Bilan : Support Vector Machine

$$\min_{\mathbf{w} \in \mathbb{R}^d} \quad ||\mathbf{w}||^2$$
$$\text{s.t.} \quad y^{(i)} \cdot \left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1$$

- problème d'optimisation quadratique sous contraintes
- objectif convexe
- problème « simple » d'un point de vue mathématique
- 20 ans de recherche pour avoir une implémentation utilisable
- aujourd'hui : plusieurs bibliothèques proposent des implémentations performantes

Et pour les données non séparables ?

- données non-séparables : $\nexists$ hpyerplan séparateur
- dans ce cas, le problème précédent n'a pas de solution
- il y a forcément au moins une contrainte violée
- critère relâché : maximiser la marge avec le moins de contraintes violées

$\Rightarrow$ SVM à marge molle

Formalisation

- pour chaque contrainte, on introduit une variable ressort :

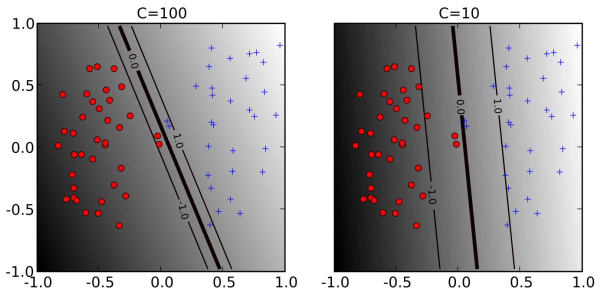
$$y^{(i)} \cdot \left(\mathbf{w} \cdot \mathbf{x}^{(i)}\right) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

- ξ_i = de combien la i^e contrainte est violée
- nouvelle objectif : maximiser la marge en minimisant le non-respect des contraintes :

$$\min ||\mathbf{w}||^2 + C \times \sum_{i=1}^{n} \xi_i$$

- C = compromis entre les deux critères = hyper-paramètres

Impact de C



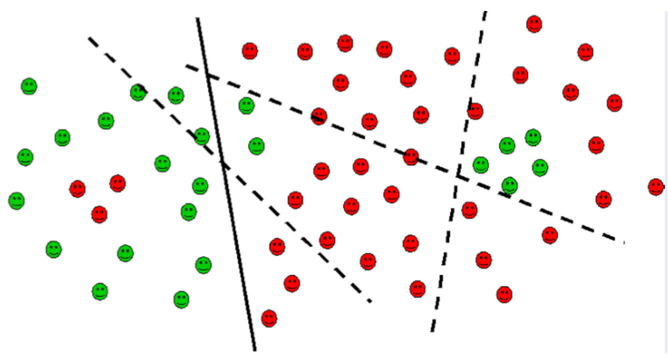
Comment choisir C

- question très importante : c'est la différence entre un SVM qui marche et un SVM qui ne marche pas
- sujet encore hautement spéculatif
- en pratique : recherche par force brute :
  - on considère un ensemble de valeurs (en général 10^i pour $i \in [\![-5, 3]\!]$)
  - on garde le C qui permet d'obtenir les meilleures performances sur un corpus de validation

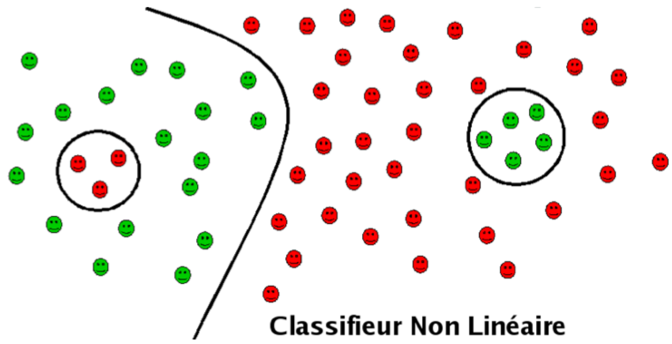
Troisième partie III

Noyaux

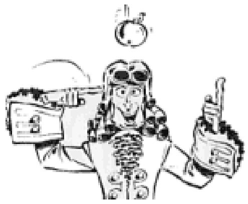
Rappel (1) : classifieurs linéaires



Rappel (2) : classifieurs non linéaires



Question locale



- Peut-on généraliser un classifieur linéaire pour apprendre une frontière non linéaire ?
- oui $\rightarrow$ noyaux
- en général associé avec les SVM, mais peut s'appliquer à tous les classifieurs linéaires

Rappel (3) : perceptron

$w \leftarrow 0$
while il y a des exemples mal classés **do**
  choisir un exemple (x_i, y_i) au hasard
  if $y_i \cdot w^\top x_i \leq 0$ **then**
    $w \leftarrow w + y_i \cdot \phi(x_i)$
  end if
end while

Representer Theorem

Observation : Toute solution trouvée par l'algorithme du perceptron est une combinaison linéaire des exemples mal classés.
(également vrai pour le SVM)

Notations :

- $w = \sum_{i=1}^n \alpha_i \cdot y_i \cdot x_i$
- $\Rightarrow$ représentation alternative du classifieur
- théorie de l'optimisation : α_i = coordonnées duales

Perceptron : forme duale

Nouvelle fonction de décision :

$$\begin{aligned} f(x) &= \text{sign}\left(\langle x | w \rangle\right) \\ &= \text{sign}\left(\left\langle \sum_{j=1}^n \alpha_j \cdot y_j \cdot x_j | x \right\rangle\right) \end{aligned}$$

Perceptron : problème d'optimisation duale

Require: a linearly separable training set $(x_i, y_i)_{i=1}^n$
  $\alpha \leftarrow 0$
  $R = \max_{i \in [\![1,n]\!]} \| x_i \|$
  while there are classification errors **do**
    for $i = 1$ to n **do**
      $y = \sum_{j=1}^n \alpha_j \cdot y_j \langle x_j | x_i \rangle$
      if $y \neq y_i$ **then**
        $\alpha_i \leftarrow \alpha_i + 1$
      end if
    end for
  end while

Intérêt

- dans la fonction de décision *et* dans l'algorithme d'optimisation, on accède aux exemples uniquement pour calculer des produits scalaires
- algorithme « rapide » si on sait calculer un produit scalaire rapidement
- et surtout : pas de dépendance quand à la dimension des exemples

Les noyaux (1)

Un noyau :

- fonction $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$...
- tel qu'il existe un espace de Hilbert $\mathcal{H}$ et une fonction $\phi : \mathcal{X} \mapsto \mathcal{H}$, avec $K(x, x') = \langle \phi(x) | \phi(x') \rangle$,

Exemple : noyau polynomial

$$\begin{aligned} K : \mathbb{R}^d \times \mathbb{R}^d &\to \mathbb{R} \\ (x, x') &\mapsto \left(\langle x | x' \rangle + 1\right)^m \end{aligned}$$

Les noyaux (2)

Supposons : $x = (x_1, x_2)$

$$\begin{aligned} K(x, x') = \left(\langle x | x' \rangle + 1\right)^2 &= \left(x_1 \cdot x_1' + x_2 \cdot x_2' + 1\right)^2 \\ &= x_1^2 \cdot x_1'^2 + \left(\sqrt{2} \cdot x_1\right) \cdot \left(\sqrt{2} \cdot x_1'\right) + \left(\sqrt{2} \cdot x_1 \cdot x_2\right) \cdot \left(\sqrt{2} \cdot x_1' \cdot x_2'\right) \\ &\quad + \left(\sqrt{2} \cdot x_2\right) \cdot \left(\sqrt{2} \cdot x_2'\right) + x_2^2 \cdot x_2'^2 + 1 \\ &= \langle \phi(x) | \phi(x') \rangle \end{aligned}$$

avec :

$$\phi : (x_1, x_2) \mapsto \begin{pmatrix} x_1^2 \\ \sqrt{2} \cdot x_1 \\ \sqrt{2} \cdot x_1 \cdot x_2 \\ \sqrt{2} \cdot x_2 \\ x_2^2 \\ 1 \end{pmatrix}$$

Bilan

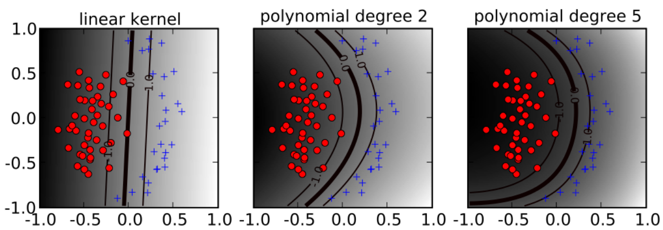
Intérêts

1. réduction de la complexité : avec un noyau $\mathcal{O}(d)$ au lieu de $\mathcal{O}\left(\binom{d+m-1}{m}\right)$
2. possibilité d'utiliser des transformations non linéaires :
   - noyaux gaussien $K(x, x') = \exp\left(-\frac{||x-x'||^2}{2 \cdot \gamma^2}\right)$

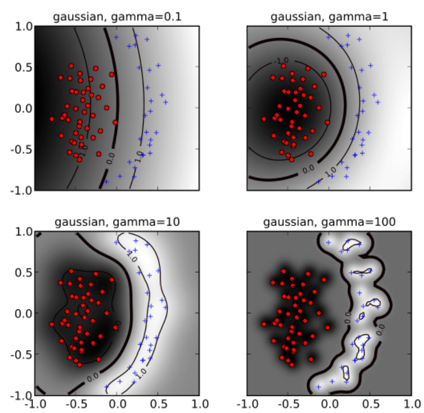
Moralité

Noyaux = moyens de réaliser une transformation non-linéaire implicite

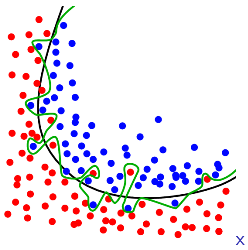
Exemple



Exemple (2)



Les noyaux une solution magique ?



- les noyaux permettent de modéliser des frontière de décisions complexes
- mais : risque fort de sur-apprentissage
- coût computationnel élevé
- peu utile en grande dimension