

# **Classification supervisée**

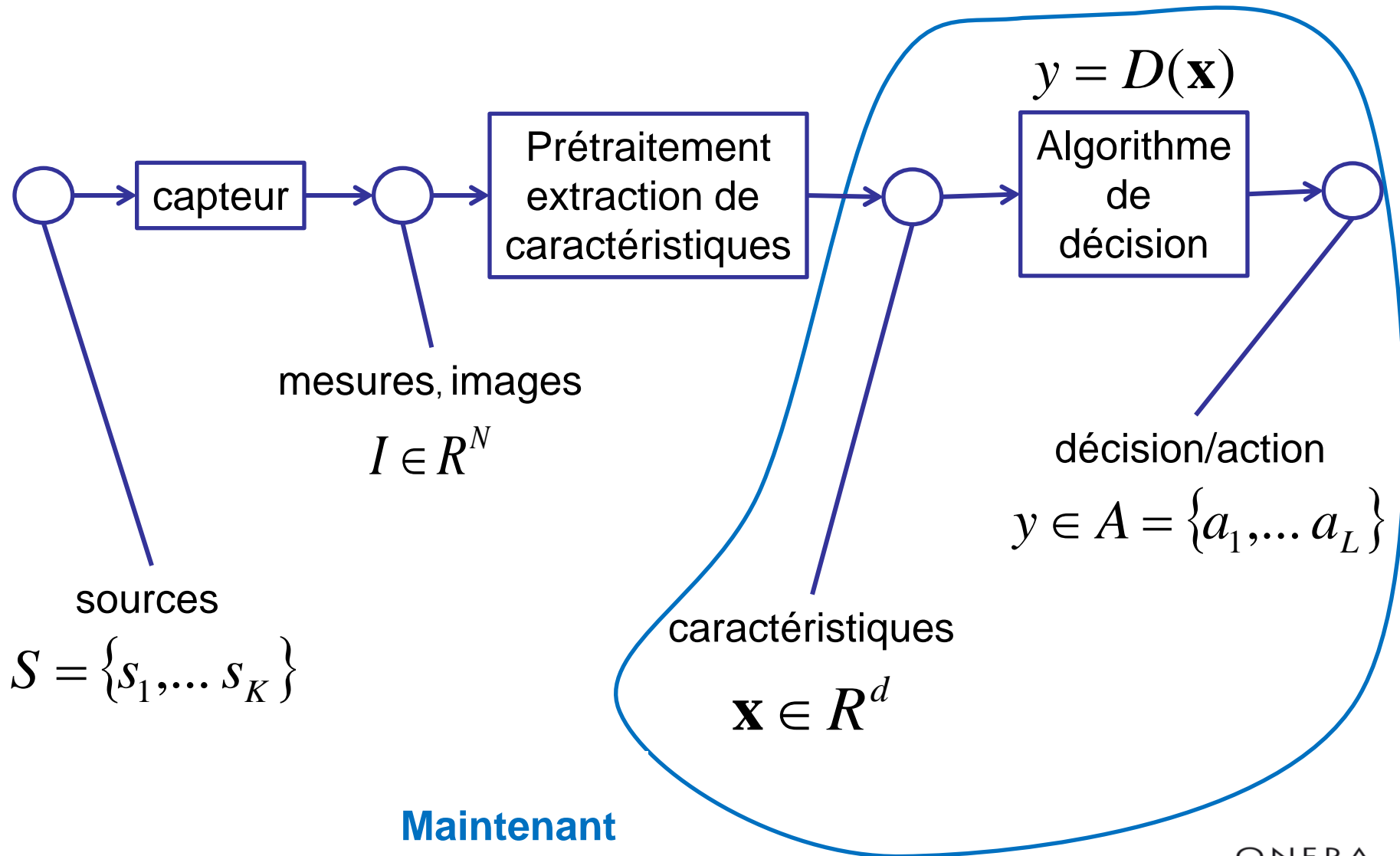
## **Exemple des « Support Vector Machines »**

**24/01/2017**

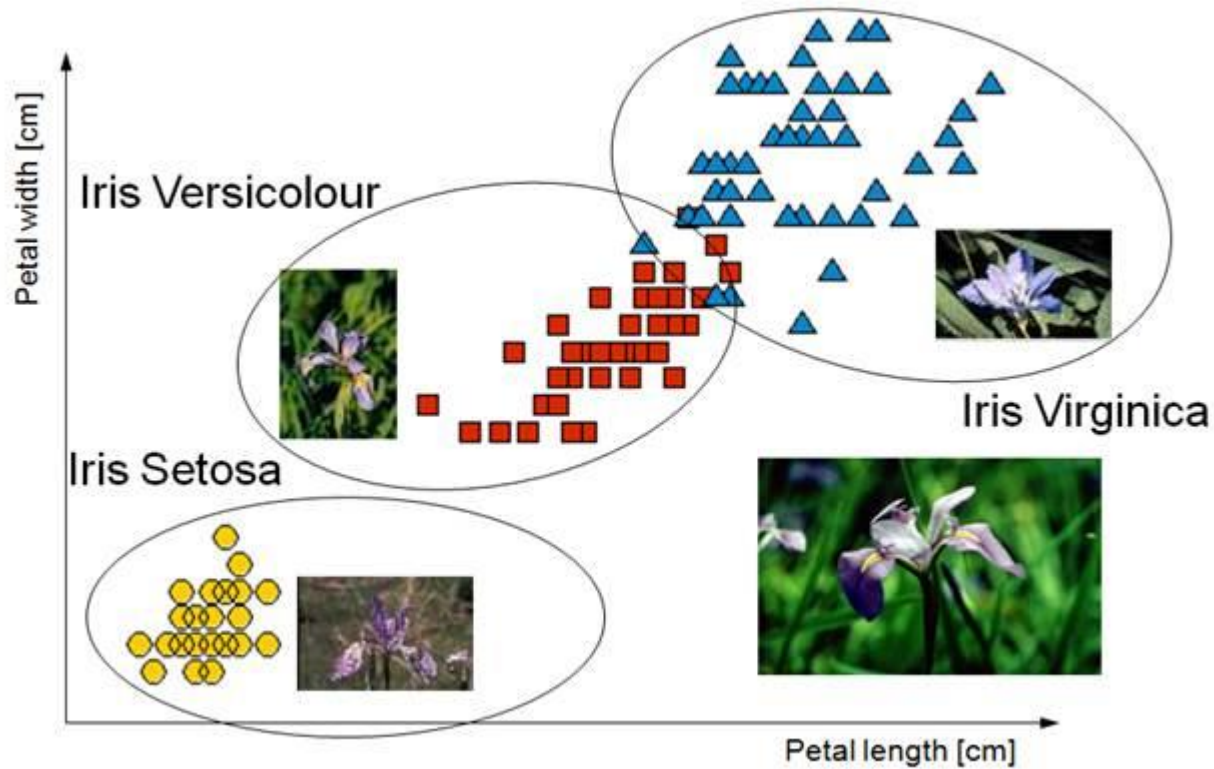
S. Herbin

DTIM/PSR

# La chaîne d'interprétation de données

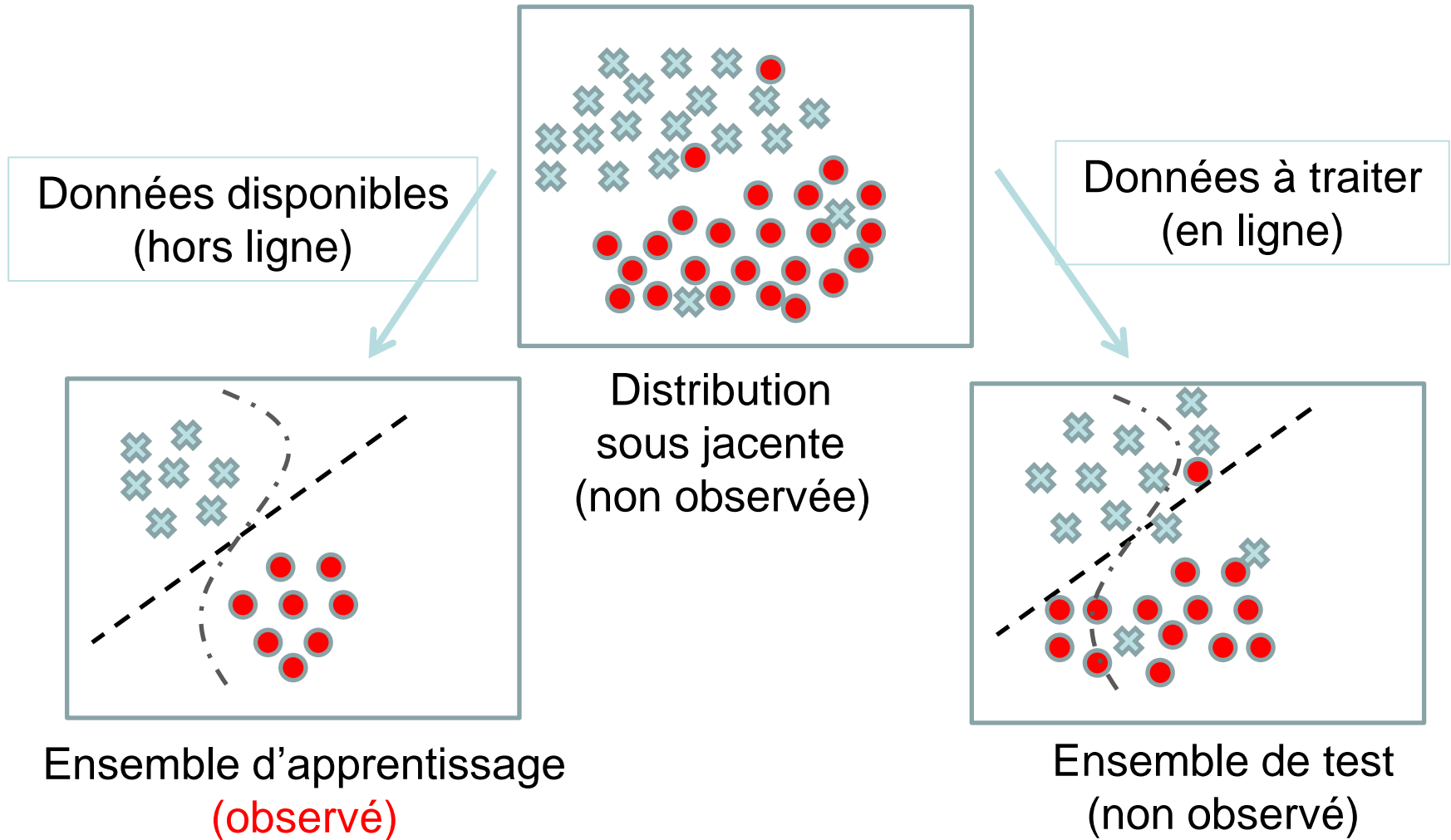


# Fonction de Décision



Partition de l'espace des caractéristiques

# Apprentissage et test



# Différents types de classification

- Binaire

$$\mathcal{A} = \{-1, 1\}$$

- Multi classe

$$\mathcal{A} = \{1, 2 \dots L\}$$

- Détection

$$\mathcal{A} = \{1, 2 \dots L\} \times R^4$$

- Caractérisation des données:

- Rejet
- Anomalie

$$\mathcal{A} = \{1, 2 \dots L, \text{ambigu}, \text{inconnu}\}$$

# Apprentissage supervisé

- On veut construire une fonction de décision  $D$  à partir d'exemples
- On dispose d'un ensemble d'apprentissage  $\mathcal{L}$  sous la forme de paires  $\{\mathbf{x}_i, y_i\}$  où  $\mathbf{x}_i$  est la donnée à classer et  $y_i$  est la vraie valeur (classe):

$$\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^N$$

- L'apprentissage consiste à trouver cette fonction de classification dans un certain espace paramétrique  $W$  optimisant un certain critère  $E$ :

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w} \in W} \mathcal{E}(\mathcal{L}, \mathbf{w})$$

- On l'applique ensuite à de nouvelles données de test:

$$y = D(\mathbf{x}; \hat{\mathbf{w}})$$

# Différentes manières de classer (cas binaire)

- Par calcul d'un score ou d'un rapport de vraisemblance puis seuil:

$$S(\mathbf{x}) > \lambda \Rightarrow y = 1$$

- Par comparaison de similarité (plus proche voisin, prototypes...)

$$y^* = \left\{ y(\mathbf{x}_i) \text{ tq } \forall j, S(\mathbf{x}, \mathbf{x}_i) < S(\mathbf{x}, \mathbf{x}_j) \right\}$$

- Par maximisation d'une fonction de compatibilité

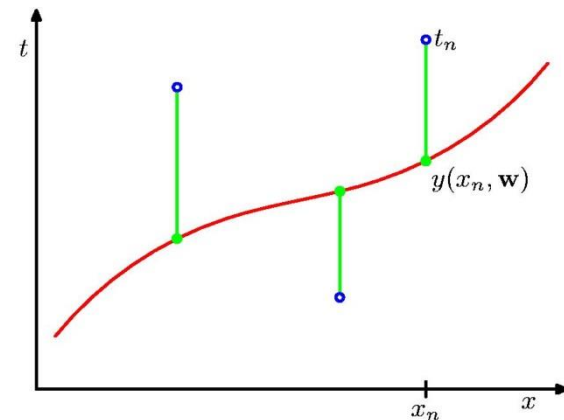
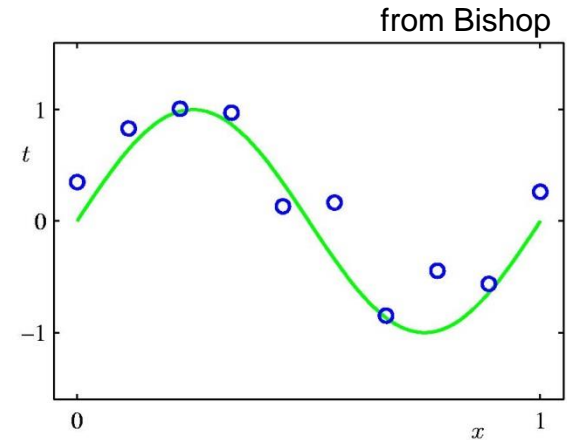
$$y^* = \arg \max(C(-1, \mathbf{x}), C(1, \mathbf{x}))$$

- ...

- Il y a (presque) toujours comme intermédiaire une ou des fonctions à valeurs réelles à calculer → Régression

# Un exemple « paradigmatique » : la régression polynomiale

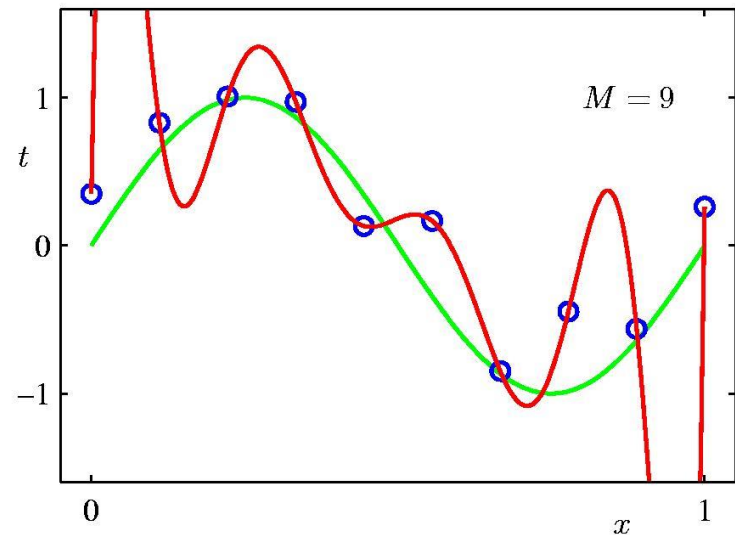
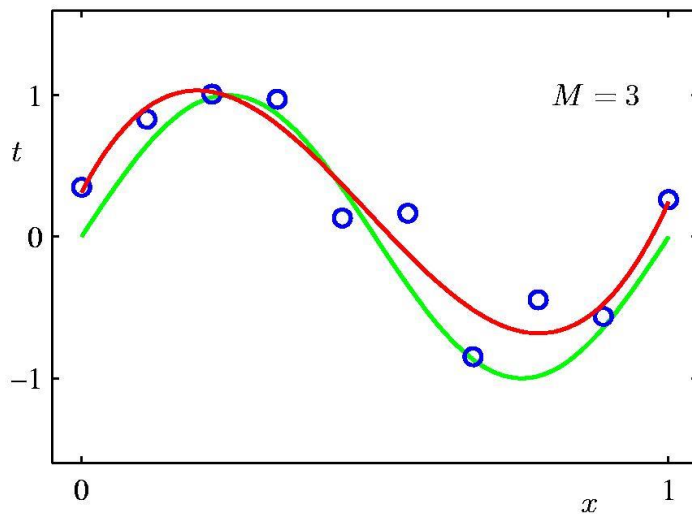
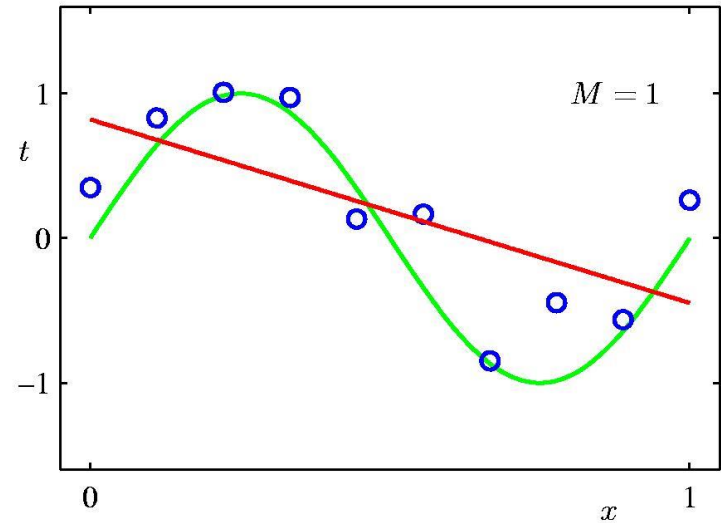
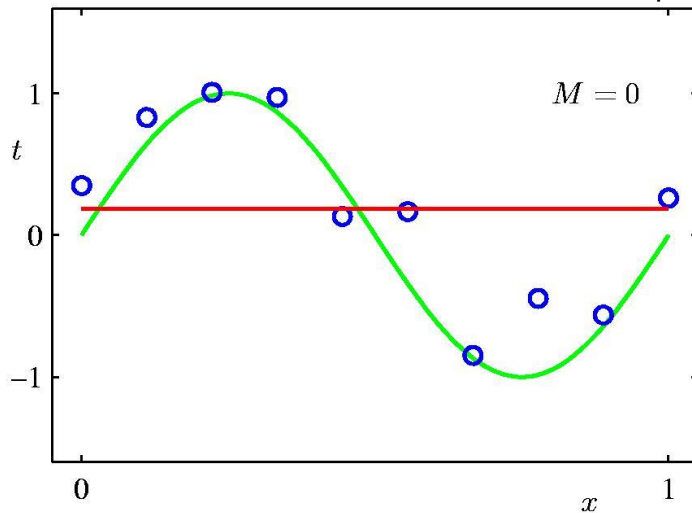
- La courbe verte est la véritable fonction à estimer (non polynomiale)
- Les données sont uniformément échantillonnées en  $x$  mais bruitées en  $y$ .
- L'erreur de régression est mesurée par la distance au carré entre les points vrais et le polynôme estimé.





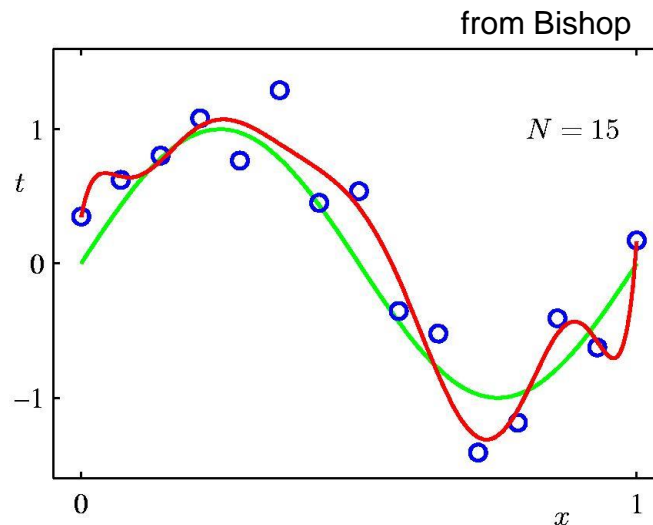
# Quelles sont les meilleures régressions?

from Bishop



# Une approche simple pour contrôler la complexité

Si on pénalise les grandes valeurs des coefficients du polynôme, on obtient une fonction moins « zigzagante »



Fonction de  
coût

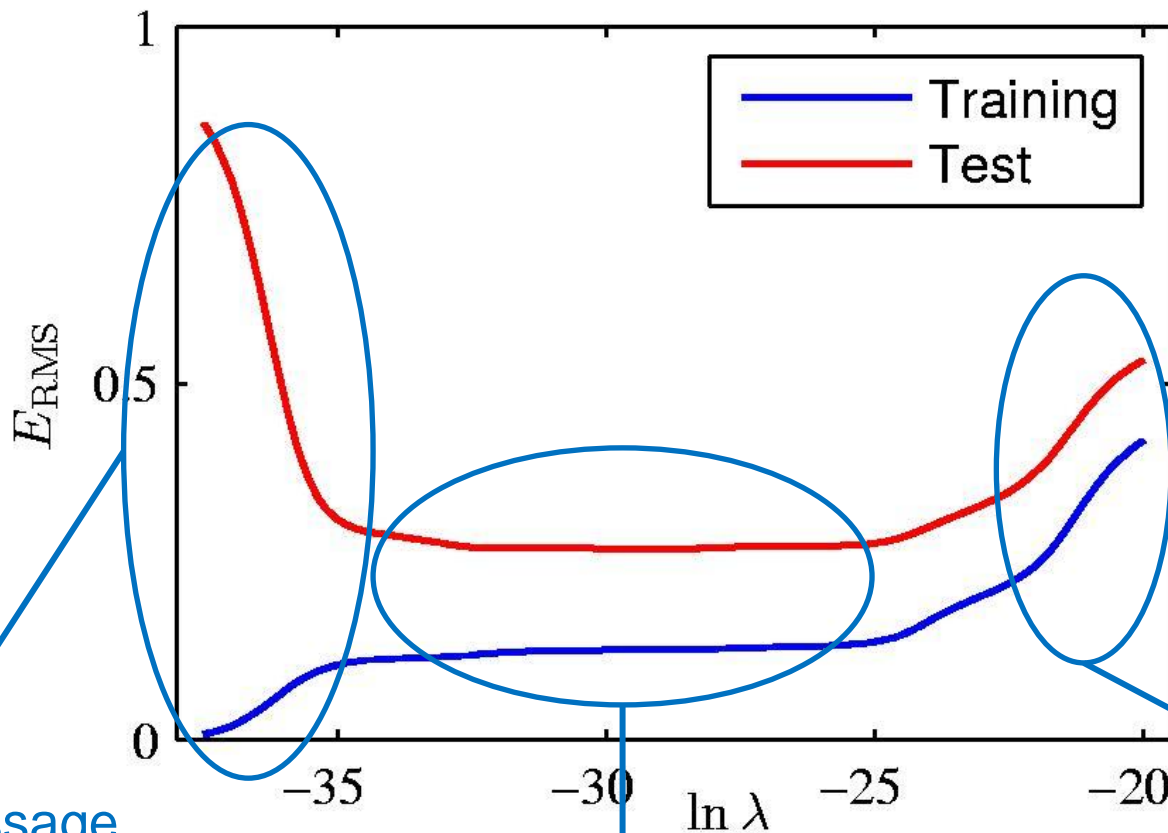
Paramètre de  
régularisation

$$\mathcal{E}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N \{ D(\mathbf{x}_i, \mathbf{w}) - t_i \}^2 + \frac{\lambda}{2} \| \mathbf{w} \|^2$$

Valeur vraie  
au point  $x_i$

# Regularisation: $\mathcal{E}_{\text{RMS}}$ vs. $\ln(\lambda)$

$$\mathcal{E}_{\text{RMS}}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N \{D(\mathbf{x}_i, \mathbf{w}) - t_i\}^2$$



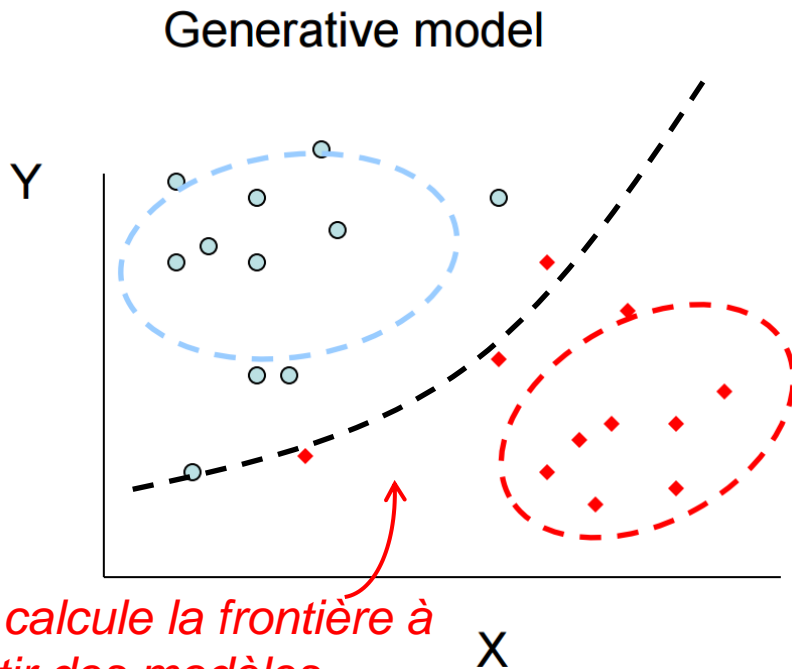
Sur apprentissage  
= « Overfitting »

Bon régime

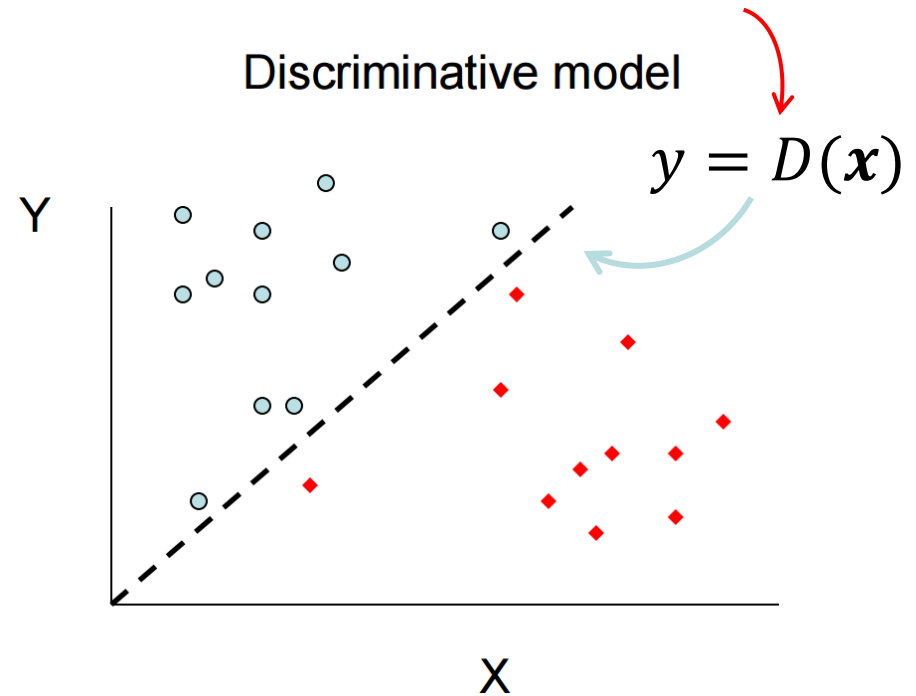
Classifieur  
pas assez expressif

# Deux types d'approches: génératives vs. discriminatives

Objectif = modéliser les distributions de données puis les exploiter



*On estime directement*



Objectif = construire les meilleures frontières

# Critères statistiques pour la classification

- Risque ou erreur empirique

$$\mathcal{E}_{\text{train}}(\mathbf{w}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N \{D(\mathbf{x}_i, \mathbf{w}) \neq y_i\}$$

- Erreur de généralisation (ou de test, ou idéale...)

$$\mathcal{E}_{\text{test}}(\mathbf{w}) = E_{\mathbf{x}, y} [\{D(\mathbf{x}, \mathbf{w}) \neq y\}]$$

- Critère à optimiser (fonction objectif):

$$\text{Loss}(\mathbf{w}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N l(D(\mathbf{x}_i, \mathbf{w}), y_i) + r(\mathbf{w})$$

Adéquation aux données

Régularisation

# Problématiques de l'apprentissage supervisé

- A distribution de données d'entrée fixée...
- Maîtriser:
  - Le choix de l'espace des classifieurs (structure et paramètres)
  - Le critère empirique à optimiser
  - La stratégie d'optimisation
  - L'évaluation de la solution
- A situer dans une démarche de conception plus globale:
  - Choix des caractéristiques
  - Temps de calcul
  - Nature de la classification
  - Performances attendues...

# Garanties théoriques (exemple)

Les données cachées

Les données disponibles

$$\boxed{\mathcal{E}_{test}} \leq \boxed{\mathcal{E}_{train}} + \left( \frac{h + h \log(2N / h) - \log(p / 4)}{N} \right)^{\frac{1}{2}}$$

Où  $N$  = nombre de données  
 $h$  = indicateur de complexité des classifieurs (VC dimension)  
 $p$  = probabilité que la borne soit fausse

En jouant sur la complexité des classes de classifieurs, on peut optimiser la borne d'erreur d'estimation.

Cette borne est plutôt lâche

En pratique, la démarche est plutôt « experte », et repose sur un certain savoir faire et une connaissance des données.

# Validation croisée

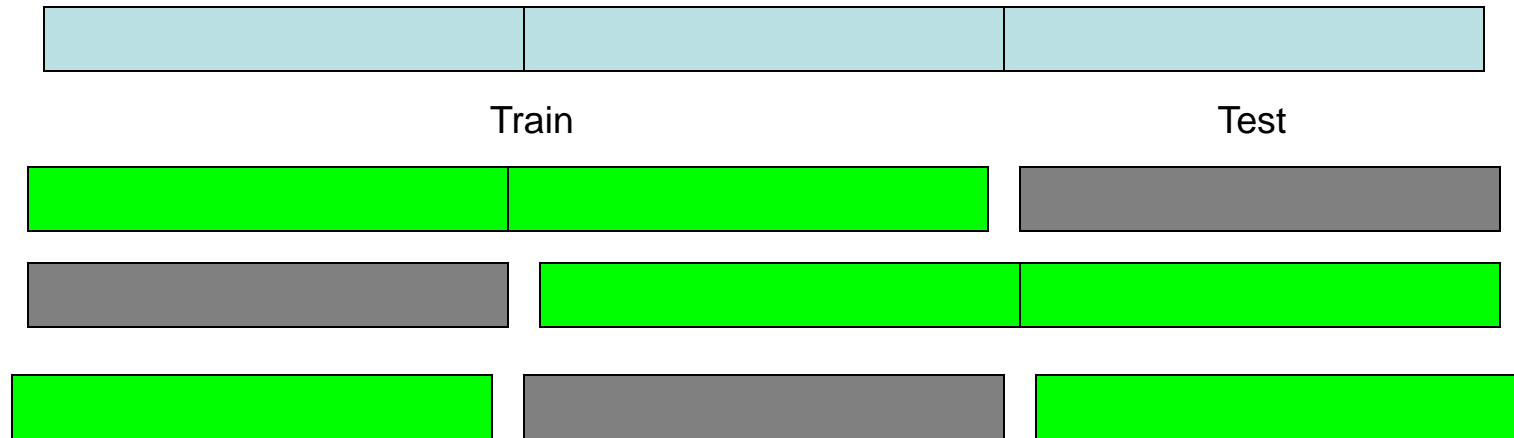
- Permet d'estimer l'erreur de généralisation à partir des données d'apprentissage (« astuce »)
- Principe:
  - Division des données en  $k$  sous ensembles (« fold »)
  - Choix d'une partie comme ensemble de *test* fictif, les autres comme *train*
  - Apprentissage sur l'ensemble *train*
  - Estimation des erreurs sur *test*
  - On fait tourner l'ensemble de *test* sur chacune des parties
  - L'erreur de généralisation estimée est la moyenne des erreurs sur chaque ensemble de *test*



# Stratégies de partitionnement

- k-fold

Données

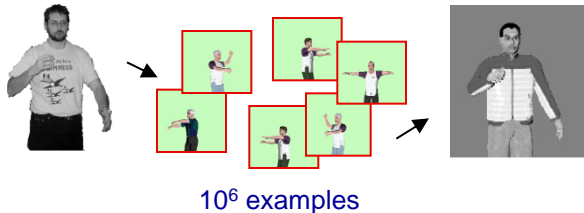


- Leave-one-out



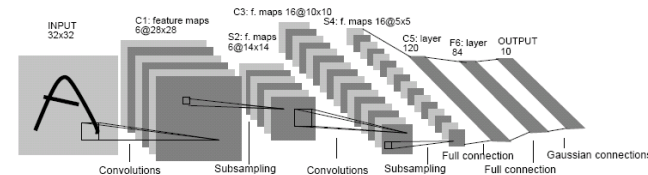
# Quelques approches discriminatives d'apprentissage supervisé en vision

## Nearest neighbor



Shakhnarovich, Viola, Darrell 2003  
Berg, Berg, Malik 2005...

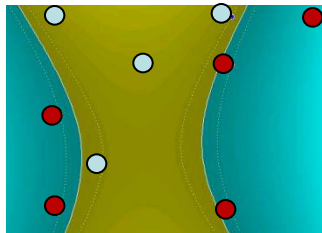
## Neural networks (Deep Learning)



LeCun, Bottou, Bengio, Haffner 1998  
Rowley, Baluja, Kanade 1998

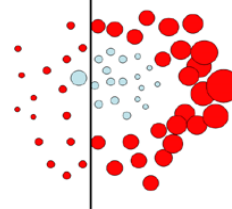
...

## Support Vector Machines



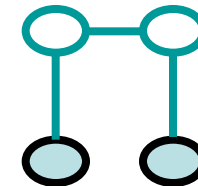
Guyon, Vapnik  
Heisele, Serre, Poggio, 2001,...

## Boosting



Viola, Jones 2001,  
Torralba et al. 2004,  
Opelt et al. 2006,...

## Conditional Random Fields



McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003  
...

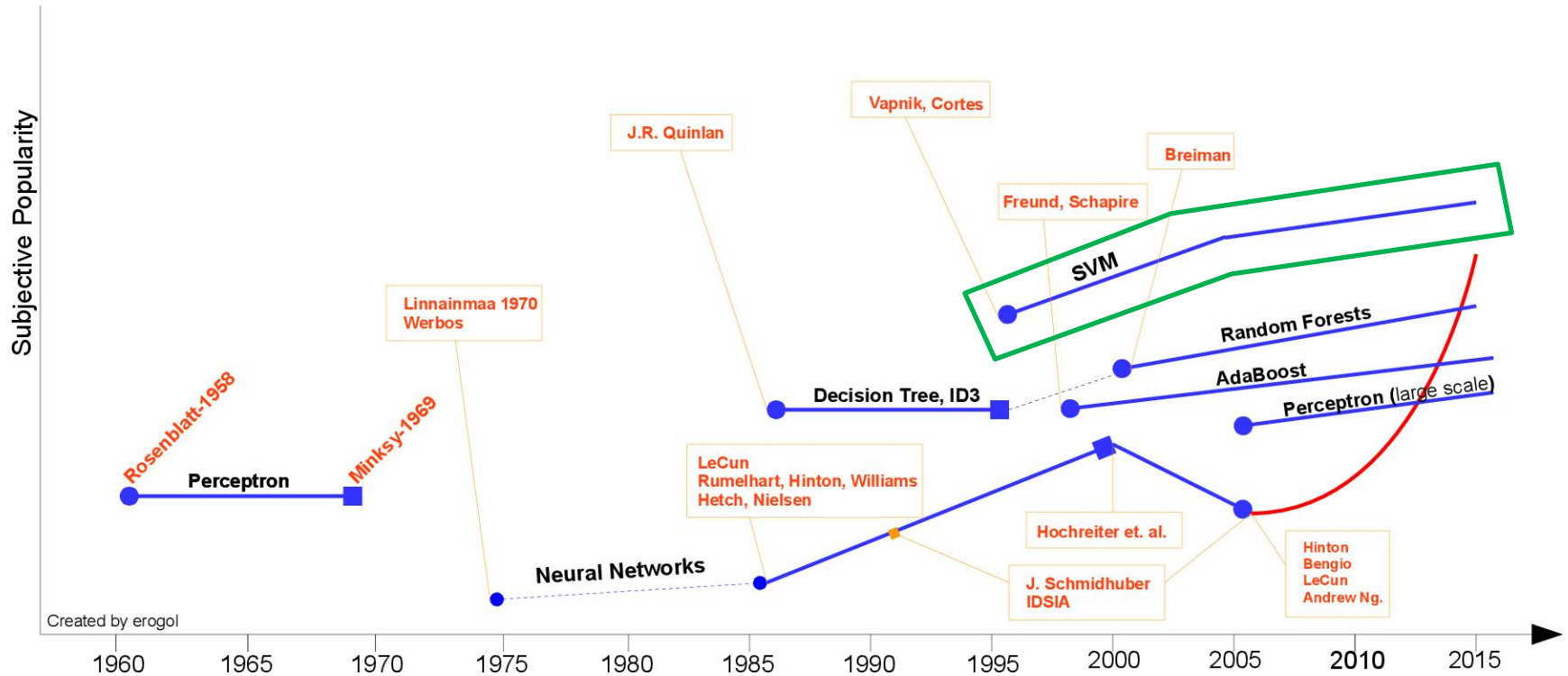
# Références

- K. Fukunaga, Introduction to Statistical Pattern Recognition (Second Edition), Academic Press, New York, 1990.
- P.A. Devijver and J. Kittler, Pattern Recognition, a Statistical Approach, Prentice Hall, Englewood Cliffs, 1982)
- R.O. Duda and P.E. Hart, Pattern classification and scene analysis, John Wiley & Sons, New York, 1973.
- L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and regression trees, Wadsworth, 1984.
- L. Devroye, L. Györfi and G. Lugosi, A Probabilistic Theory of Pattern Recognition, (Springer-Verlag 1996)
- S. Haykin, Neural Networks, a Comprehensive Foundation. (Macmillan, New York, NY., 1994)
- V. N. Vapnik, The nature of statistical learning theory (Springer-Verlag, 1995)
- C. Bishop, Pattern Recognition and Machine Learning, (Springer-Verlag, 2006).
- Jerome H. Friedman, Robert Tibshirani et Trevor Hastie, The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Springer-Verlag 2009).
- Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning, An MIT Press book (<http://www.deeplearningbook.org>)

# Support Vector Machines

- Historique
- Principe: maximiser la marge de séparation d'un hyperplan
- Le cas séparable
- Le cas non séparable: les fonctions de perte (« hinge loss »)
- L'extension au cas non linéaire: les noyaux
- Le calcul effectif
- Les paramètres de contrôle

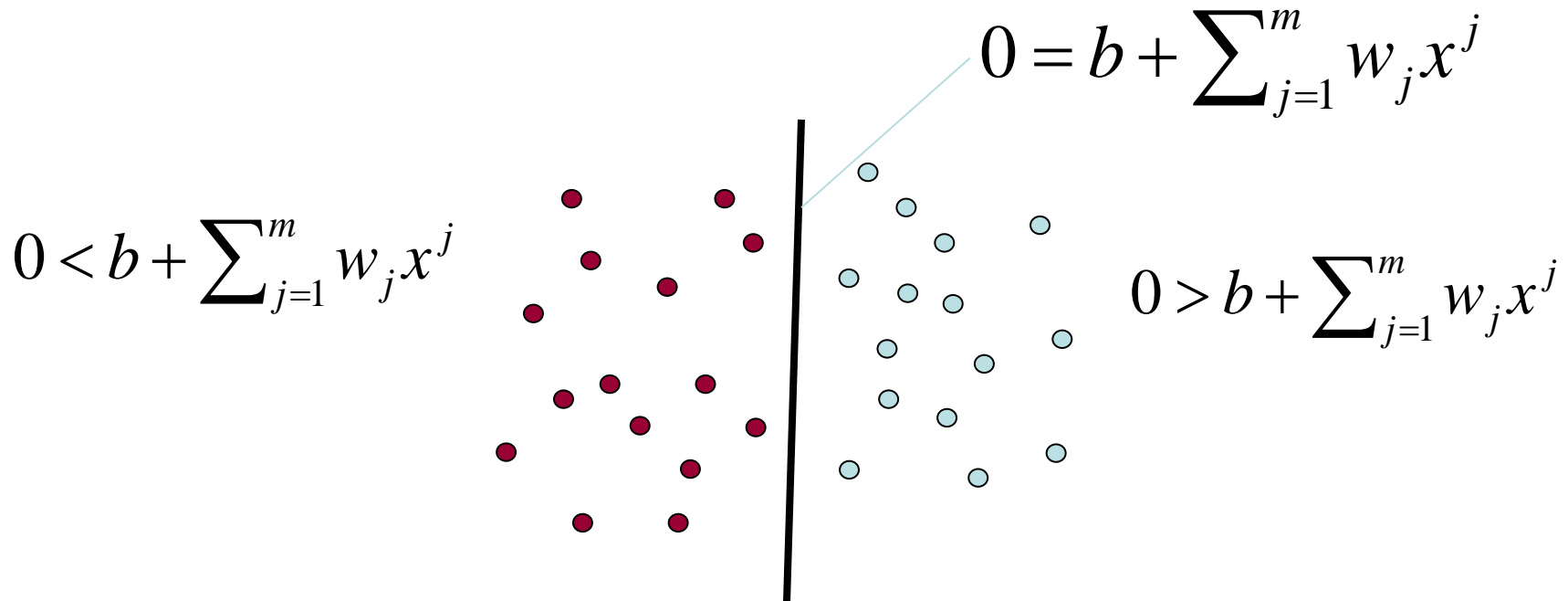
# Historique du Machine Learning



# Modèles linéaires de décision

Hypothèse = les données sont linéairement séparables.

- En 2D, par une droite
- En ND, par un hyperplan.



# Classifieur linéaire

- Equation de l'hyperplan séparateur

$$b + \mathbf{w} \cdot \mathbf{x} = 0$$

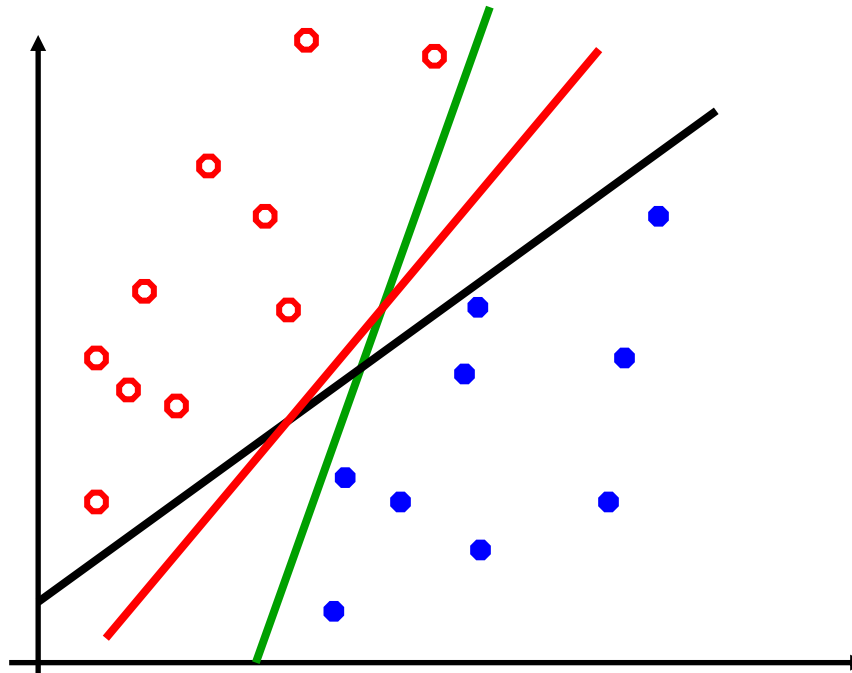
- Expression du classifieur linéaire (pour  $y_i$  valant -1 et 1)

$$D(\mathbf{x}; \mathbf{w}) = \text{sign}(b + \mathbf{w} \cdot \mathbf{x})$$

- Erreur

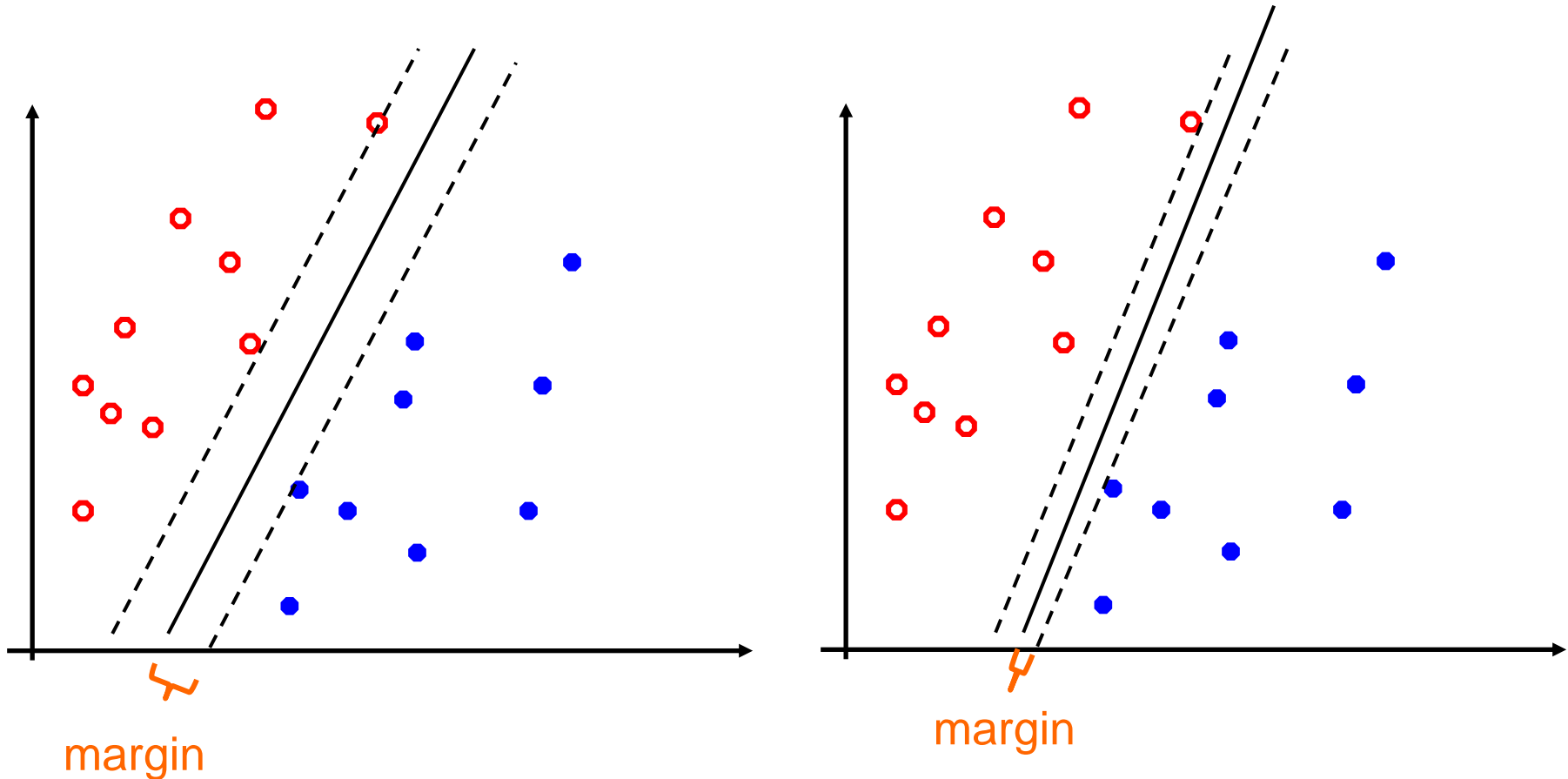
$$\mathcal{E}_{test}(\mathbf{w}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N \{y_i \cdot \text{sign}(b + \mathbf{w} \cdot \mathbf{x}_i) < 0\}$$

# Quel hyperplan choisir?





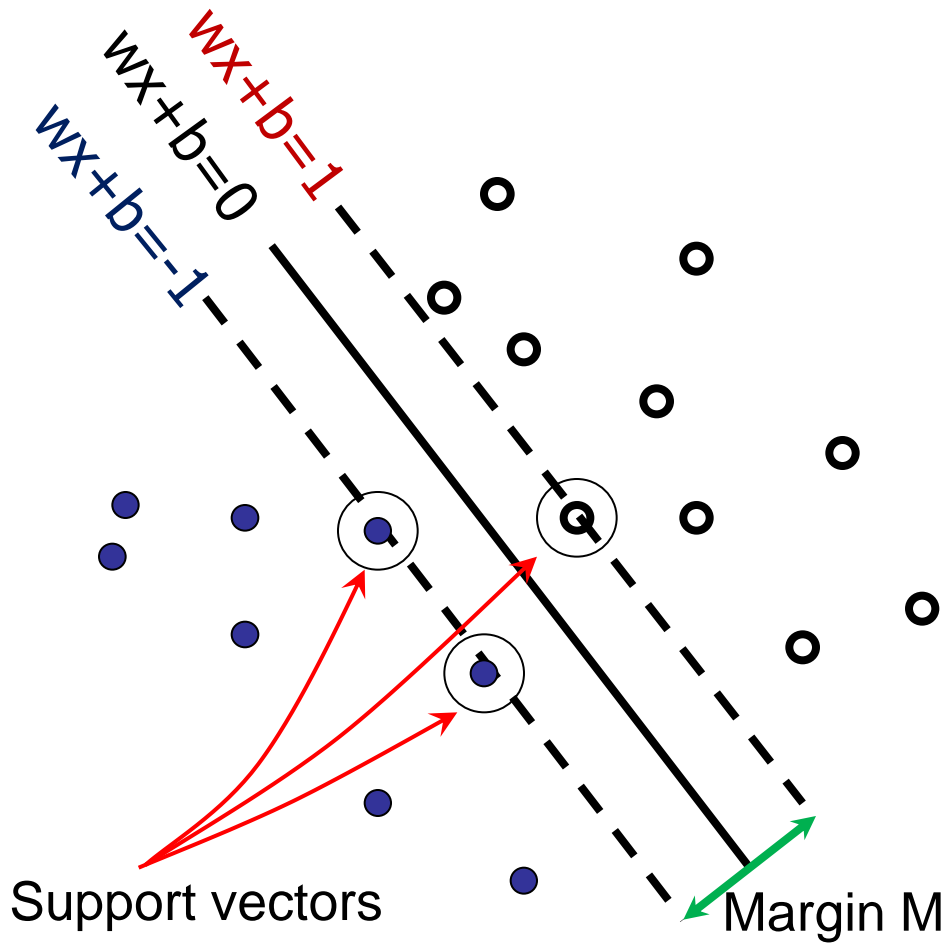
# Classifieur « Large margin »



Choisir l'hyperplan qui maximise la distance  
aux points les plus proches

# Support Vector Machines

- On cherche l'hyperplan qui maximise la marge.



$$\mathbf{x}_i \text{ positif } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ négatif } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

Pour les vecteurs de support,  $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance entre point et hyperplan:

$$\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$$

Pour les « support vectors »:

$$\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \quad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

# Principe du SVM (Large Margin)

- Maximiser la marge = distance des vecteurs à l'hyperplan séparateur des vecteurs de supports

$$\max \frac{1}{\|\mathbf{w}\|^2}$$

- Sous contraintes

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \forall i$$

- Les vecteurs de support vérifiant:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$$

Le 1 est conventionnel.  
N'importe quelle  
constante >0 est valable.

# Formulation du SVM

$$\min_{w,b} \|w\|^2$$

Tel que:

$$y_i (w \cdot x_i + b) \geq 1 \quad \forall i$$

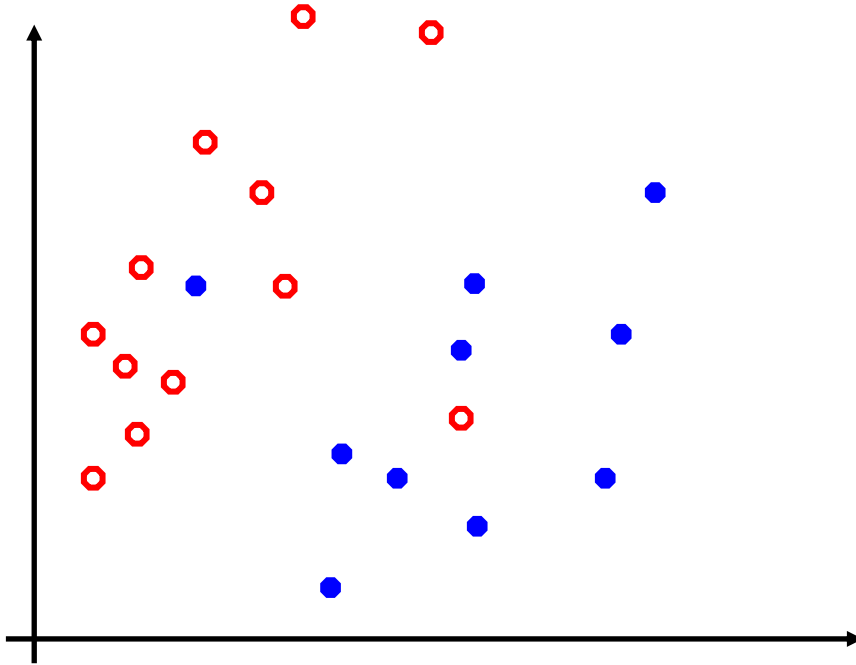
Si les données sont séparables

Problème d'optimisation quadratique

Avec contraintes linéaires

➔ Grand nombre de manières de l'optimiser!

# Classification « Soft Margin »



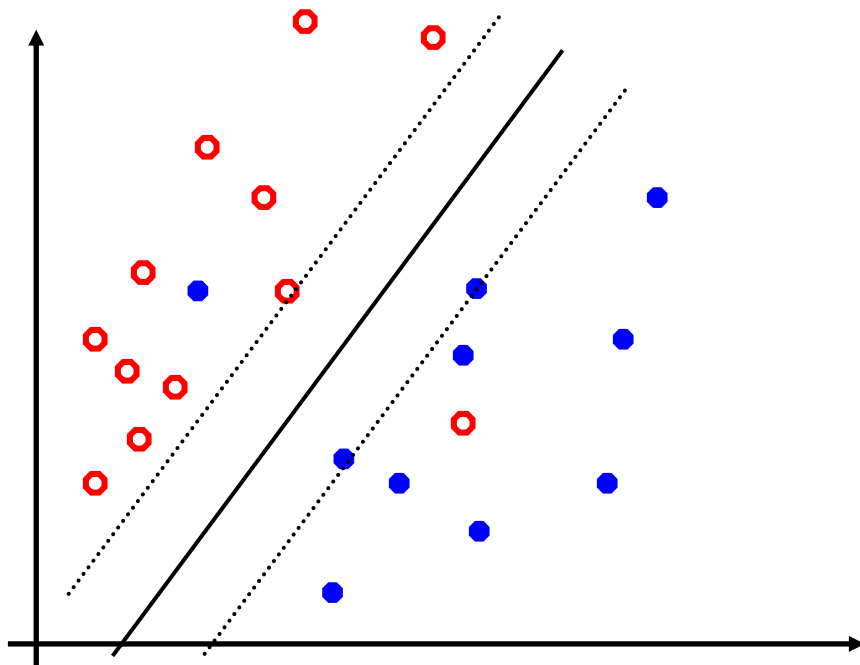
$$\min_{w,b} \|w\|^2$$

Tel que:

$$y_i (w \cdot x_i + b) \geq 1 \quad \forall i$$

Comment traiter le cas non linéairement séparable?

# Classification « Soft Margin »



$$\min_{w,b} \|w\|^2$$

Tel que:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

On aimerait obtenir une séparation robuste à quelques données non séparées

# Idée: « Slack variables »

$$\min_{w,b} \|w\|^2$$

tq:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$



$$\min_{w,b} \|w\|^2 + C \sum_i \zeta_i$$

tq:

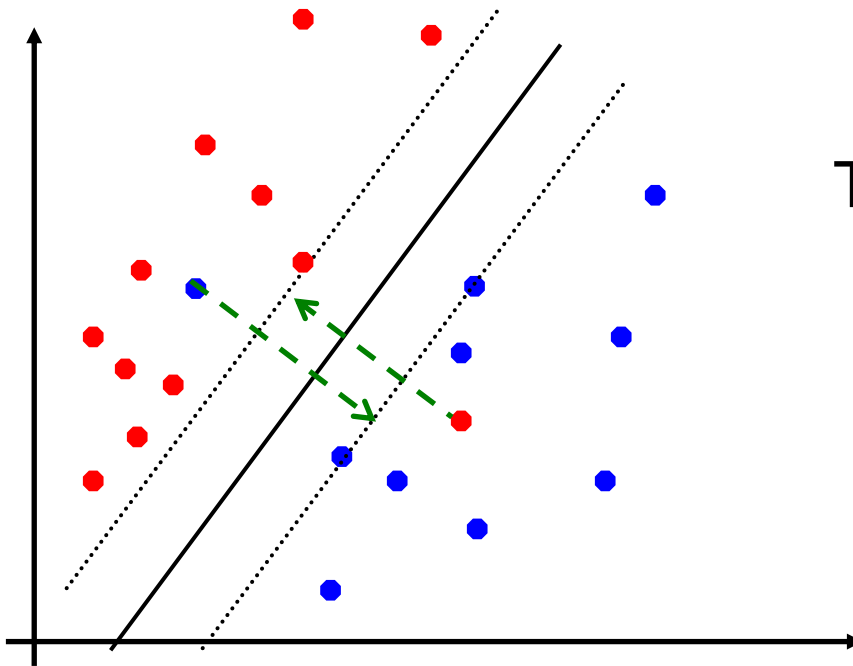
$$y_i(w \cdot x_i + b) \geq 1 - \zeta_i \quad \forall i$$

$$\zeta_i \geq 0$$

**Permet de relacher la contrainte de séparabilité pour chaque exemple.**

slack variables  
(une par exemple)

# « Slack variables »



$$\min_{w,b} \|w\|^2 + C \sum_i \varsigma_i$$

Tel que:

$$y_i(w \cdot x_i + b) + \varsigma_i \geq 1 \quad \forall i$$

$$\varsigma_i \geq 0$$

## Relâchement de la contrainte



# Utilisation des « Slack variables »

marge

Compromis entre marge et  
pénalisation de la contrainte

$$\min_{w,b} \|w\|^2 + C \sum_i \varsigma_i$$

Valeur du  
relâchement de la  
contrainte

tq

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$

Contrainte autorisée  
à être relâchée

# Soft margin SVM

$$\min_{w,b} \|w\|^2 + C \sum_i \zeta_i$$

Tel que

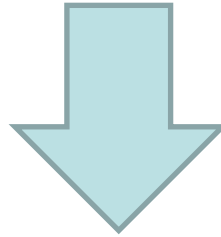
$$y_i(w \cdot x_i + b) \geq 1 - \zeta_i \quad \forall i$$
$$\zeta_i \geq 0$$

**On garde un problème quadratique!**

# Autre formulation

tq:

$$\min_{w,b} \|w\|^2 + C \sum_i \varsigma_i$$
$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$
$$\varsigma_i = \max(0, 1 - y_i(w \cdot x_i + b))$$



$$\min_{w,b} \|w\|^2 + C \sum_i \max(0, 1 - y_i(w \cdot x_i + b))$$

**Problème d'optimisation non contraint**

**→ Autres méthodes d'optimisation (descente de gradient)**

# Interprétation du « Soft Margin SVM »

$$\min_{w,b} \|w\|^2 + C \sum_i \max(0, 1 - y_i(w \cdot x_i + b))$$

On retrouve la formulation:

$$\text{Loss}(\mathbf{w}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N l(D(\mathbf{x}_i, \mathbf{w}), y_i) + r(\mathbf{w})$$

Avec

$$r(\mathbf{w}) = \frac{1}{C} \|\mathbf{w}\|^2$$

$$l(D(\mathbf{x}_i, \mathbf{w}), y_i) = \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b))$$

**Le SVM est un cas particulier du formalisme:  
« erreur empirique + régularisation »**

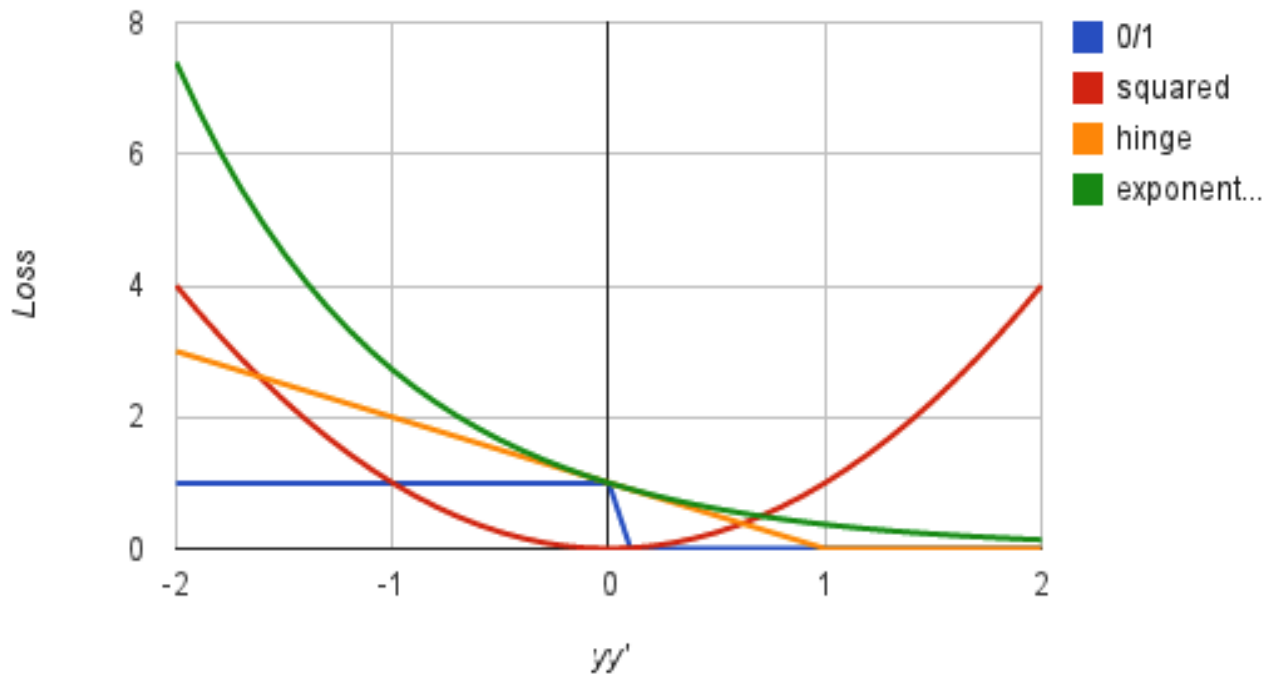
# Autres Fonctions de coût

0/1 loss:  $l(y, y') = 1[y y' \leq 0]$

Hinge:  $l(y, y') = \max(0, 1 - y y')$

Squared loss:  $l(y, y') = (y - y')^2$  Exponential:  $l(y, y') = \exp(-y y')$

Surrogate loss functions



# Forme duale du SVM

- Problème d'optimisation sous contrainte

*Pour simplifier l'expression des calculs*

Primal

$$\operatorname{argmin}_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2} + C \sum_i \xi_i \quad \text{Multiplicateurs de Lagrange}$$

$$s. t. \quad \forall i, y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad \alpha_i$$

$$\xi_i \geq 0 \quad \beta_i$$

Dual (Lagrangien)

$$L(\mathbf{w}, \xi, \alpha, \beta)$$

$$= \frac{\|\mathbf{w}\|^2}{2} + \sum_i (C\xi_i - \alpha_i(y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i) - \beta_i\xi_i)$$

$$s. t. \quad \forall i, \alpha_i \geq 0, \beta_i \geq 0$$

# Forme duale du SVM

- Lagrangien

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j \cdot \mathbf{x}_i$$

Maximisation dans le dual!

$$s. t. \forall i, 0 \leq \alpha_i \leq C$$

On garde un  
pb. quadratique

Dual des contraintes « slack »

Solution optimale (conditions de Kuhn-Tucker):  $\alpha_i (y_i w^T x_i - 1 + \xi_i) = 0$

Interprétation:  $\alpha_i = 0$  si la contrainte est satisfaite (bonne classification)

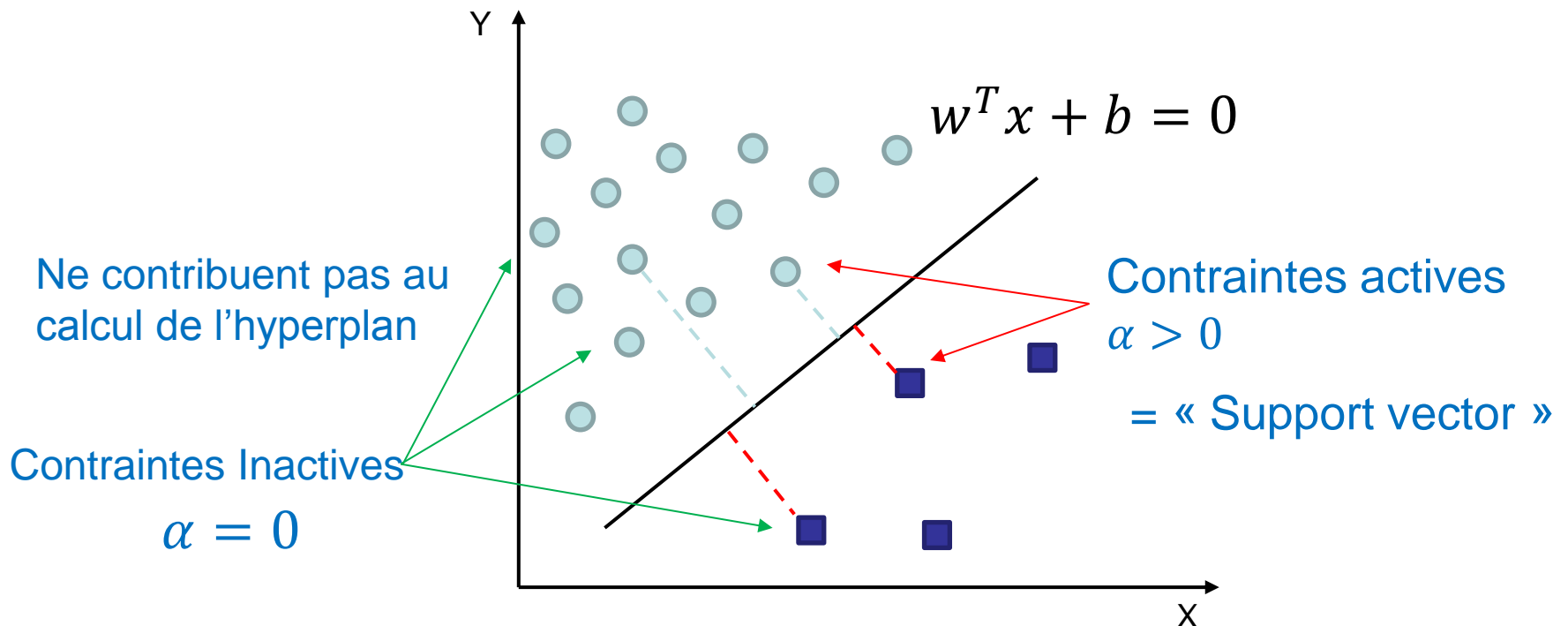
$\alpha_i > 0$  si la contrainte n'est pas satisfaite (mauvaise classification)

# Sparsité du SVM

- Seuls certains  $\alpha$  sont non nuls = autre manière de définir les vecteurs de support.

$$\text{Optimalité} = \alpha_i (y_i w^T x_i - 1 + \xi_i) = 0$$

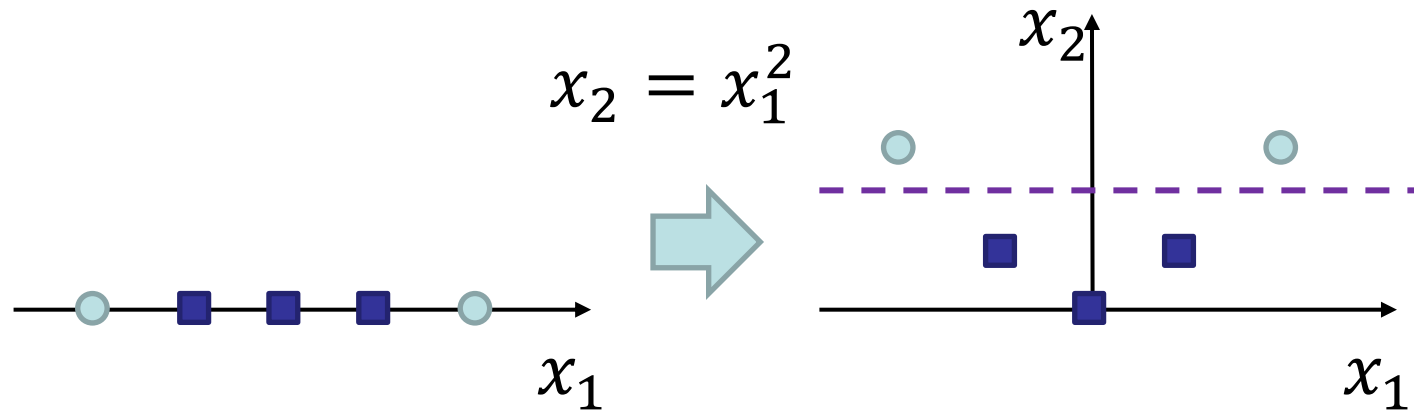
Direction de l'hyperplan séparateur  $w = \sum_i \alpha_i y_i x_i$





# Données non linéairement séparables

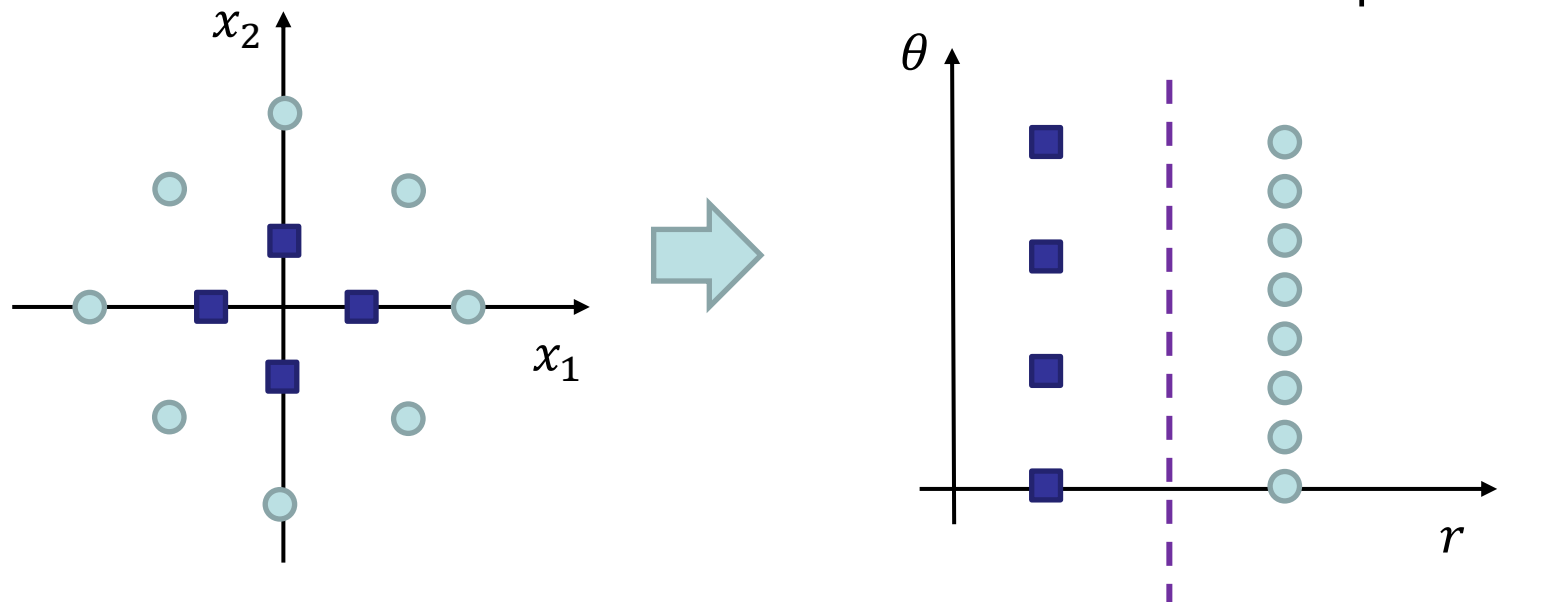
- Transformation non linéaire  $\phi(x)$  pour séparer linéairement les données d'origine



$\phi(x)$  = Transformation polynomiale

# Données non linéairement séparables

- Transformation non linéaire  $\phi(x)$  pour séparer linéairement les données d'origine



$\phi(x)$  = Transformation polaire

# Retour sur la formulation duale du SVM

Lagrangien

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \boxed{x_i x_j}$$

$$\text{tq } \forall i, 0 \leq \alpha_i \leq C$$

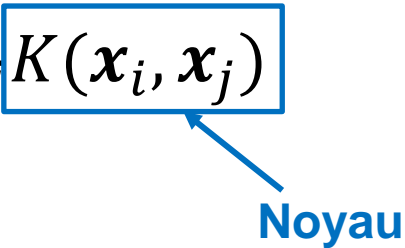
**Produit scalaire  
uniquement**



# « Kernel trick »

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

tq  $\forall i, 0 \leq \alpha_i \leq C$



Noyau

Le noyau  $K$  est un produit scalaire dans l'espace transformé:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Il est uniquement nécessaire de connaître la similarité entre données pour introduire la non linéarité dans le problème (avec des conditions...)

# Utilisation de noyaux dans les SVM

- Permet d'introduire des mesures de similarités propres au domaine étudié et sans avoir à gérer la complexité de la transformation
- Permet de séparer modélisation = noyau de la classification et SVM (optimisation)
- Définit la fonction de classification à partir de noyaux « centrés » sur les vecteurs de support

$$D(\mathbf{x}, \mathbf{w}) = b + \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x})$$

# Noyaux courants

- Polynômes de degrés supérieurs à  $d$

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d$$

- Noyau gaussien

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{(\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y})}{2\sigma^2}\right)$$

Paramètres à définir  
= degré de liberté  
supplémentaire

- Intersection d'histogrammes

$$K(\mathbf{x}, \mathbf{y}) = \sum_i \min(x^i, y^i)$$

# Intérêt de la formalisation par noyaux

- Noyaux = partie « métier »
  - Connaissance du sens à donner à la similarité
  - Choix expert des espaces de représentation et des caractéristiques informatives
- Classification/Optimisation = partie « ML »
  - Utilisation d'optimiseurs génériques
  - Certaine optimalité à représentation du problème donnée
- Utilisation des SVM à noyaux.
  - Représentations multi dimensionnelles (mais pas trop...)
  - Données de volume raisonnable
  - mais passage à l'échelle plus délicat (on se restreint alors en général aux noyaux linéaires)
- « Deep Learning » montrera que ce peut être sous-optimal...

# Codes

- LibSVM + LibLinear (nombreuses interfaces)

<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

<https://www.csie.ntu.edu.tw/~cjlin/liblinear/>

- Scikit-Learn (Python)

<https://www.csie.ntu.edu.tw/~cjlin/liblinear/>

- Matlab



# Résumé sur SVM

- Une formulation optimale quadratique du problème de classification binaire:
  - Primal: optimisation d'un critère empirique + régularisation
  - Dual: permet d'introduire sparsité et « kernel trick »→ plusieurs manières d'optimiser
- Les solutions s'expriment comme des combinaisons linéaires éparses de noyaux:

$$D(\mathbf{x}, \mathbf{w}) = b + \sum_i \alpha_i y_i \mathbf{K}(\mathbf{x}_i, \mathbf{x})$$

où  $\alpha_i > 0$  seulement pour les vecteurs de support, 0 sinon.

- En pratique, ce qu'il faut régler:
  - Le coefficient de régularisation: C
  - Le type de noyau et ses caractéristiques
  - Les paramètres de l'optimiseur

# Résumé

- Apprentissage supervisé
  - Un problème ancien, avec un corpus théorique solide (statistique) et des outils pratiques (optimiseurs, environnements logiciels)
  - Mais rien ne vaut une bonne modélisation du problème (quoique → DL)
- SVM
  - Une démarche assez générique et solide théoriquement
  - Performances plutôt bonnes pour jeux de données modestes, et faibles dimensions de représentation
- Il y a d'autres approches (arbres de décision, boosting, bagging, random forrest...) qui ont d'autres qualités (espaces de caractéristiques et de décision structurés ou plus complexes, gestion des données manquantes, incrémentalité...)