

Master IA2S (Intelligence Artificielle, Science des données et Systèmes Cyber-Physiques)

Apprentissage Automatique

Ferhat ATTAL

ferhat.attal@u-pec.fr

Université Paris Est Créteil (UPEC)

Octobre 2021

Cours 3 : Apprentissage supervisé

Apprentissage supervisé

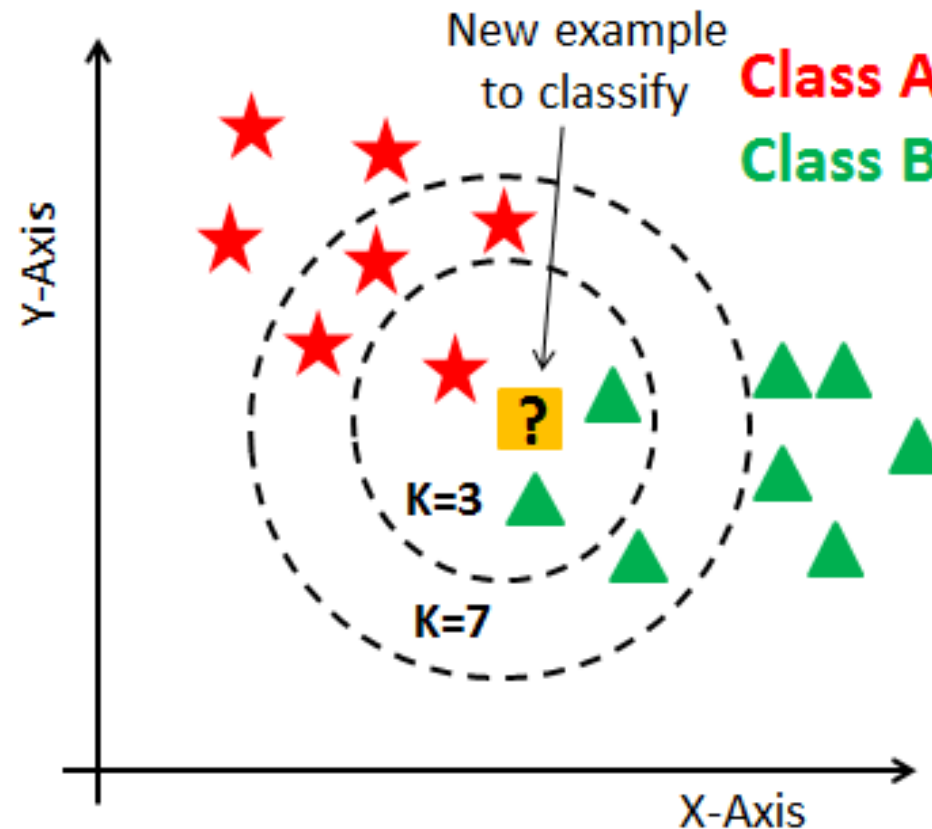
- k -plus proches voisins
- Naïve bayes
- Analyse discriminante linéaire
- Séparateurs à Vaste Marge
- Arbre de décision
- Forêts aléatoires
- Réseau de neurones artificiels

***k*-plus proches voisins**

- Généralités
- Éléments clés dans la mise en œuvre de l'algorithme des *k*-ppv
- Algorithme
- Exemple

Généralités

L'algorithme des k -plus proches voisins (ou k -nn de l'anglais k -nearest neighbors algorithm) [Cover and Hart, 1967], [Aha, 1992] est l'un des algorithmes d'apprentissage automatique les plus simple. Cet algorithme est considéré comme l'un des meilleurs algorithmes de fouille de données ([Wu et al., 2008]) pour sa capacité à produire des classifieurs simples mais puissants



Généralités

- Malgré sa simplicité, l'algorithme k -ppv donne souvent des résultats compétitifs.
- L'algorithme k -ppv est un algorithme de la famille des algorithmes dits « paresseux » (il n'y a pas de processus de construction de modèle)
- Le processus d'apprentissage dans le cas des k -ppv est simplement une question de regroupement des données d'apprentissage selon leurs étiquettes.
- L'algorithme k -ppv est basé sur le principe de similarité des propriétés entre l'ensemble d'apprentissage et les nouvelles réalisations à classer ([Cover and Hart, 1967]).

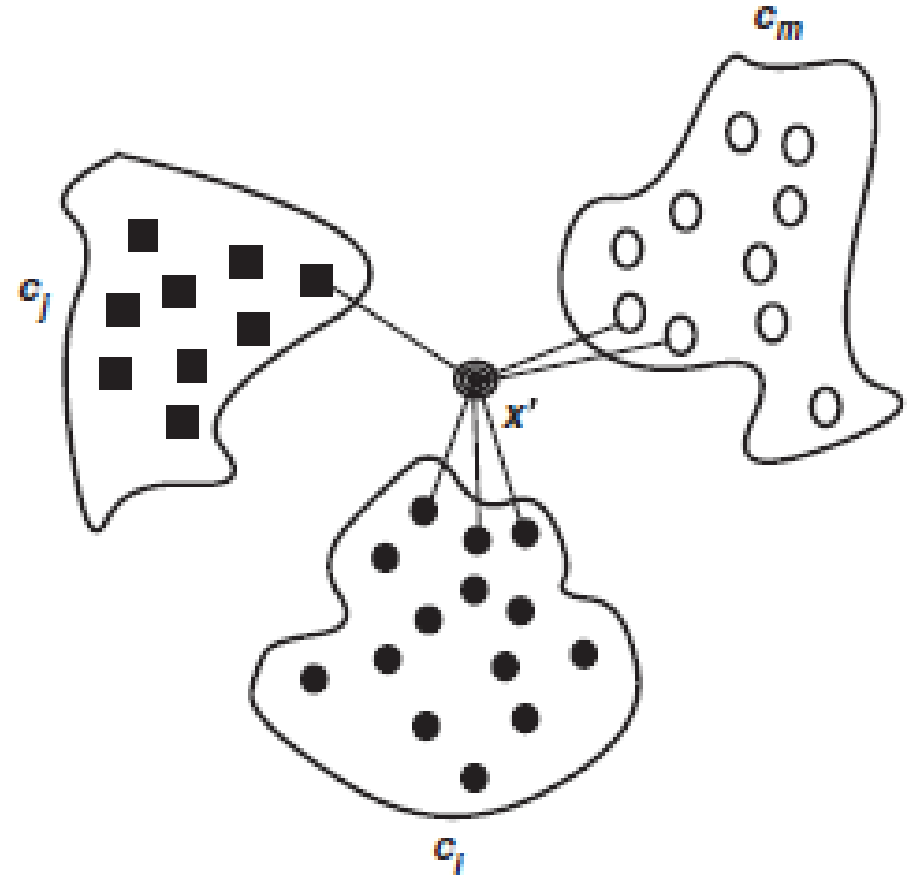
Éléments clés dans la mise en œuvre de l'algorithme des k -ppv

- Un ensemble d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ où la variable d'entrée x_i appartient à \mathbb{R}^d et la variable d'intérêt y_i appartient à un ensemble fini à K classes $Y = \{c_1, c_2, \dots, c_K\}$
- Une distance ou similitude d pour calculer la distance entre les objets,
- La valeur de k , qui représente le nombre de voisins les plus proches.

Éléments clés dans la mise en œuvre de l'algorithme des k -ppv

La règle de classification par les k -ppv est la plus simple de celles appliquées dans la reconnaissance des formes. Pour classer un nouveau objet x_{new} non étiqueté on calcule la distance de cet objet par rapport aux objets étiquetés x_i .

Exemple de classification basé sur la règle de k plus proches voisins ($k=6$). Dans cet exemple, la classe majoritaire est c_i (3 voisins)



Éléments clés dans la mise en œuvre de l'algorithme des k -ppv

Les distances les plus utilisées sont :

La distance Euclidienne

$$d(x_{new}, x_i) = \sqrt{\sum_{j=1}^d (x_{new}^j - x^j)^2}$$

La distance de Manhattan

$$d(x_{new}, x_i) = \sum_{j=1}^d |x_{new}^j - x^j|$$

La distance de Minkowski

$$d(x_{new}, x_i) = \sqrt[q]{\sum_{j=1}^d |x_{new}^j - x^j|^q}$$

L'algorithme k -ppv

Algorithm 1 .

Inputs : L'ensemble de données d'apprentissage $S = \{(x_i, y_i)\}_{i=1}^n$

Une nouvelle instance x' à classifier

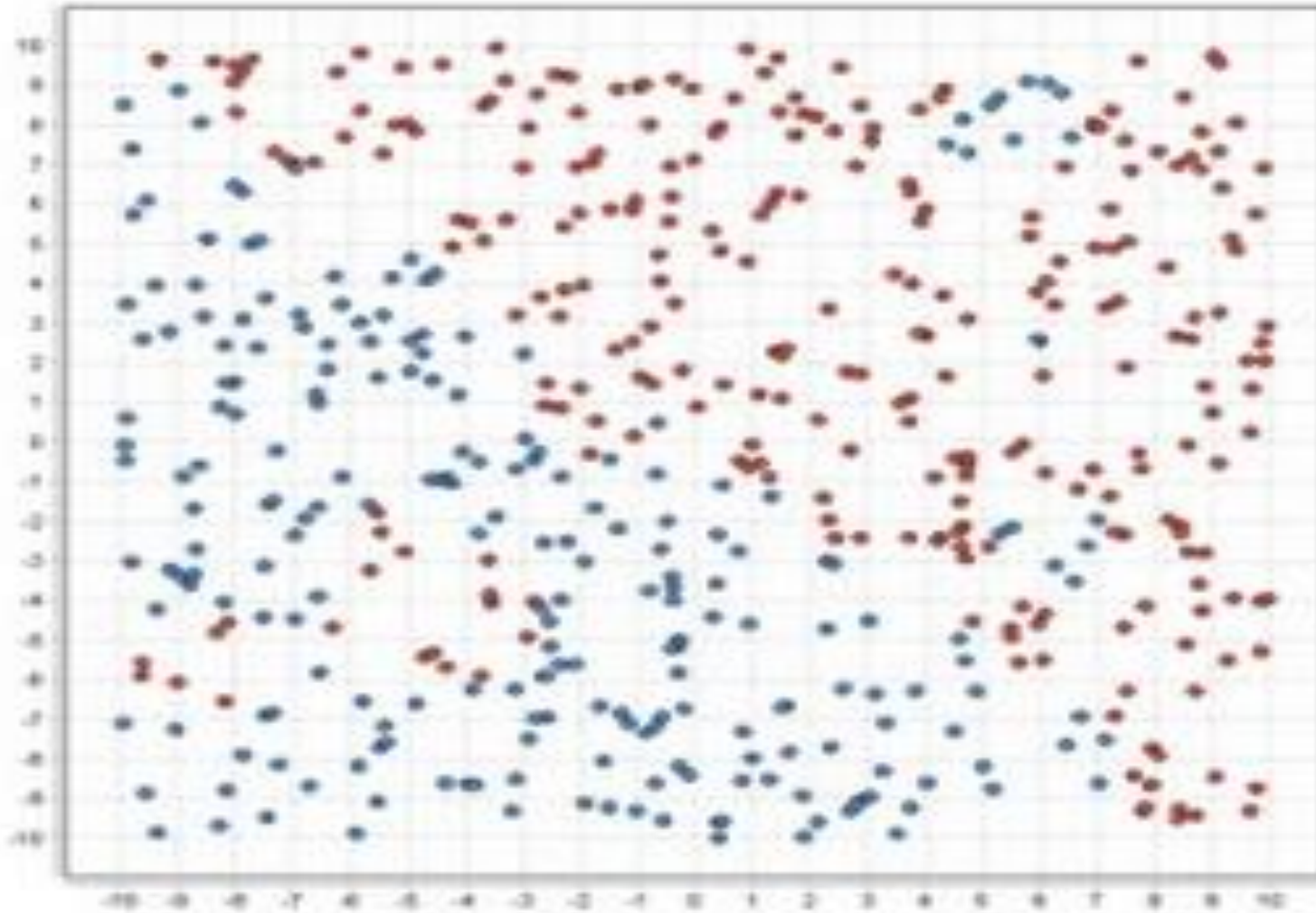
Nombre de plus proche voisins k

- 1: **for** $i=1, \dots, n$ **do**
- 2: calculer la distance $d(x', x_i)$ entre x_i et x'
- 3: **end for**
- 4: sélectionner les k distances les plus proches de x'
- 5: affecter x' à la classe majoritairement représentée par ses voisins

Outputs : \hat{y} . la classe de x' selon les k plus proches voisins

Example

Data



Exemple

$k=1$



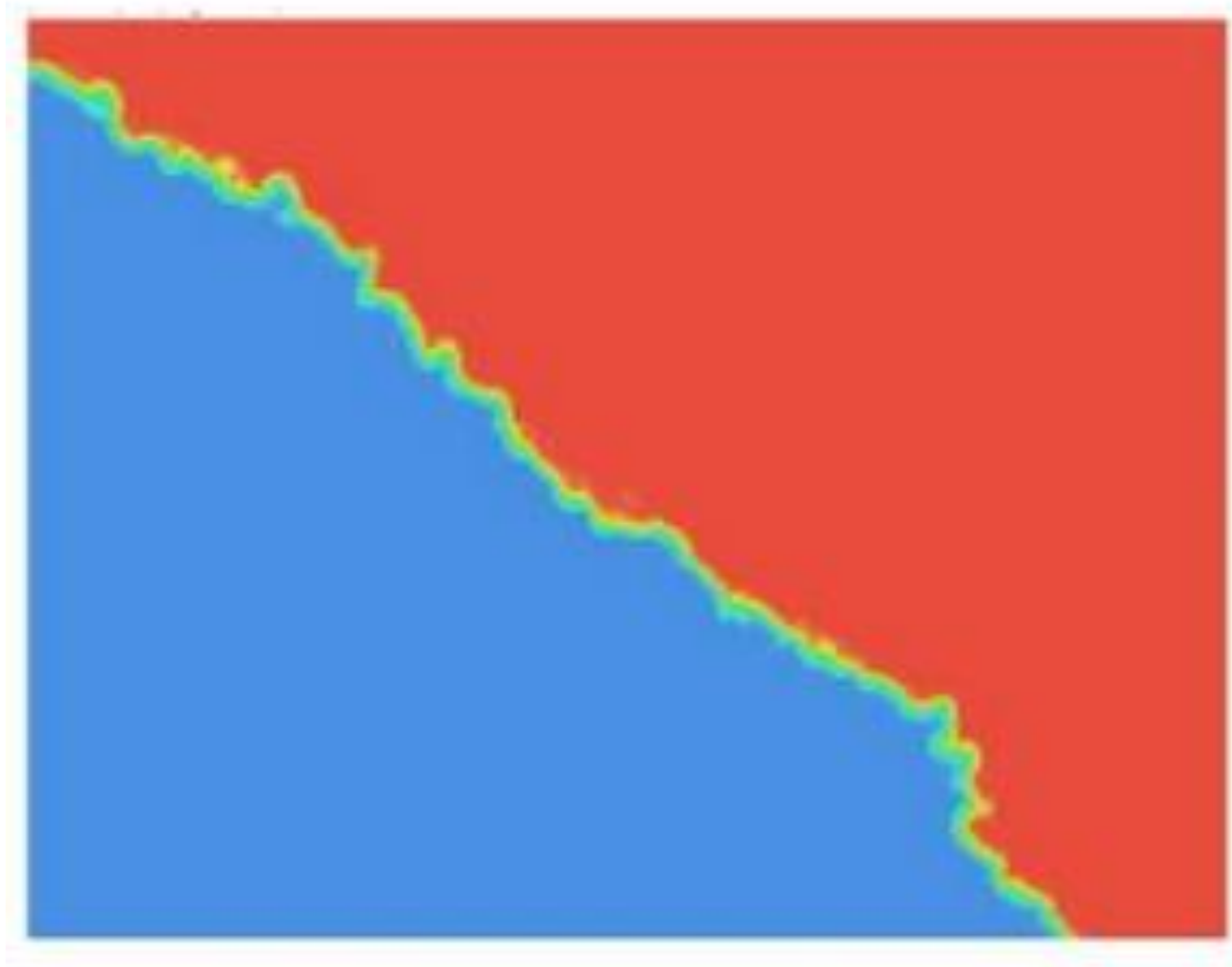
Exemple

$k=10$



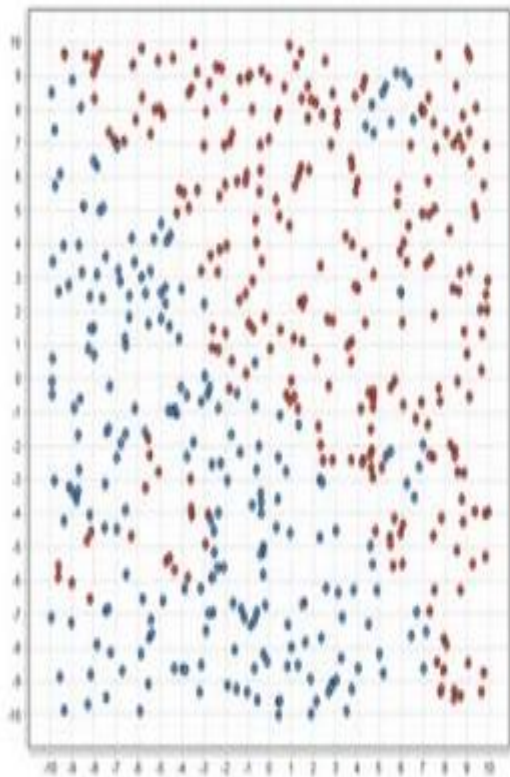
Example

$k=50$

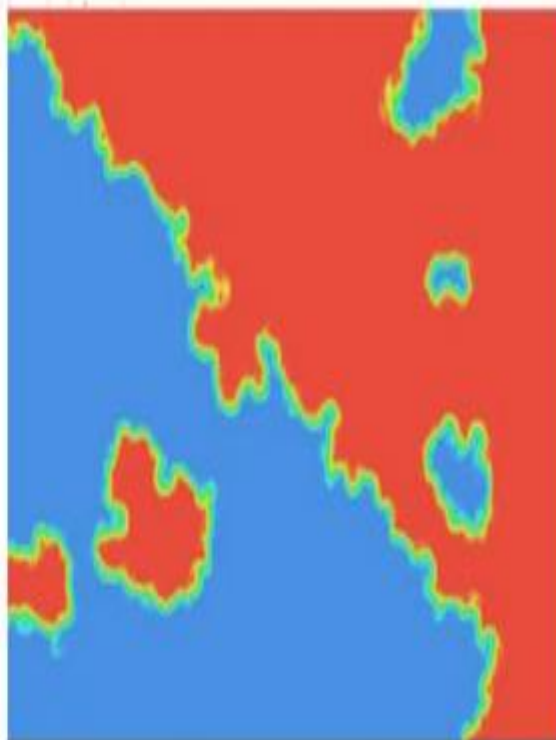


Example

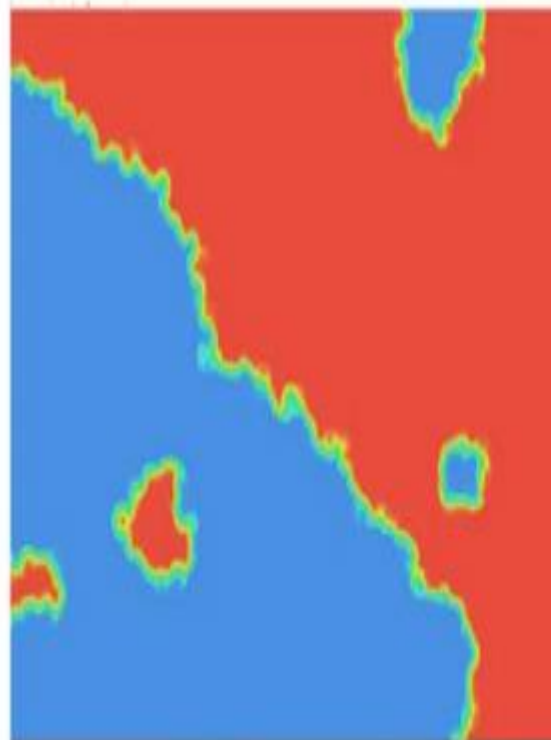
Data



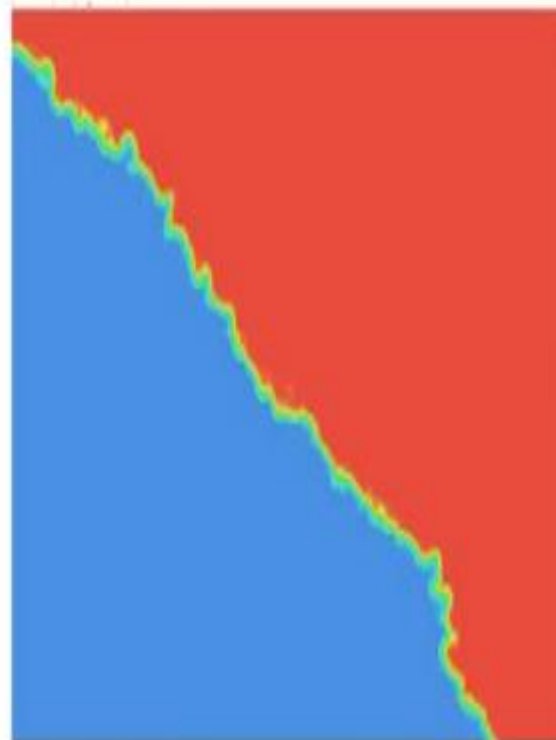
$k=1$



$k=10$



$k=50$



Exemple

sepal.length	sepal.width	petal.length	petal.width	variety
4.6	3.2	1.4	0.2	Setosa
5.3	3.7	1.5	0.2	Setosa
5	3.3	1.4	0.2	Setosa
7	3.2	4.7	1.4	Versicolor
6.4	3.2	4.5	1.5	Versicolor
6.9	3.1	4.9	1.5	Versicolor
7.2	3	5.8	1.6	Virginica
7.4	2.8	6.1	1.9	Virginica
7.9	3.8	6.4	2	Virginica
4.7	3.2	1.3	0.2	?

Naïve bayes

- Introduction
- Théorème de Bayes
- Application à la classification
- Estimation des paramètres du modèle Naïve bayes
 - Cas continu
 - Cas discret
- Exemples

Introduction

Le naïve de Bayes est un modèle d'apprentissage supervisé basé sur le théorème de Bayes avec de fortes hypothèses d'indépendance entre les variables d'entrées. Le principal avantage de ce modèle réside dans la simplicité de sa mise en œuvre qui ne nécessite aucune estimation itérative compliquée des paramètres



Thomas Bayes

Théorème de Bayes

Le théorème de Bayes fournit un moyen de calculer la probabilité à postériori $P(A|B)$ à partir de $P(A)$, $P(B)$ et $P(B|A)$.

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

- $P(A|B)$: probabilité à postériori, la probabilité de notre hypothèse A étant donné B
- $P(B|A)$: la vraisemblance de l'événement B .
- $P(A)$: Probabilité à priori A .
- $P(B)$: Probabilité marginale de B .

Application à la classification

Soit un ensemble d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ où la variable d'entrée x_i appartient à \mathbb{R}^d et la variable d'intérêt y_i appartient à un ensemble fini à K classes, $Y = \{c_1, c_2, \dots, c_k, \dots, c_K\}$

$$P(y_i = c_k | x_i) = \frac{P(y_i = c_k)P(x_i | y_i = c_k)}{P(x_i)}$$

On sait que

$$\begin{aligned} P(y_i = c_k)P(x_i | y_i = c_k) &= P(y_i = c_k, x_i) \\ &= P(y_i = c_k, x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}) \end{aligned}$$

Application à la classification

$$\begin{aligned} &P(y_i = c_k, x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}) \\ &= P(y_i = c_k)P(x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}|y_i = c_k) \\ &= P(y_i = c_k)P(x_{i1}|y_i = c_k, x_{i2}, \dots, x_{il}, \dots, x_{id})P(x_{i2}, \dots, x_{il}, \dots, x_{id}|y_i = c_k) \\ &= \\ &\quad \cdot \\ &\quad \cdot \\ &\quad \cdot \\ &= P(y_i = c_k)P(x_{i1}|y_i = c_k, x_{i2}, \dots, x_{il}, \dots, x_{id}) \dots P(x_{i(d-1)}|y_i = c_k, x_{id}) P(x_{id}|y_i = c_k) \end{aligned}$$

L'hypothèse naïve: On suppose que chaque x_{il} est indépendant des autres x_{ij} pour tout $l \neq j$.

On obtient

$$P(x_{il}|y_i = c_k, x_{i(l+1)}, \dots, x_{id}) = P(x_{il}|y_i = c_k)$$

Application à la classification


Ainsi

$$\begin{aligned} P(y_i = c_k, x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}) &= P(y_i = c_k) P(x_{i1}|y_i = c_k) P(x_{i2}|y_i = c_k) \dots P(x_{il}|y_i = c_k) \dots P(x_{id}|y_i = c_k) \\ &= P(y_i = c_k) \prod_{l=1}^d P(x_{il}|y_i = c_k) \end{aligned}$$

On a également

$$\begin{aligned} P(x_i) &= \sum_{k=1}^K P(y_i = c_k, x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}) \\ &= \sum_{k=1}^K P(y_i = c_k) \prod_{l=1}^d P(x_{il}|y_i = c_k) \end{aligned}$$

Application à la classification

$$P(y_i = c_k | x_i) = \frac{P(y_i=c_k)P(x_i|y_i=c_k)}{P(x_i)}$$
$$= \frac{P(y_i=c_k) \prod_{l=1}^d P(x_{il}|y_i=c_k)}{\sum_{k=1}^K P(y_i=c_k) \prod_{l=1}^d P(x_{il}|y_i=c_k)}$$


Peut être ignorée dans la prédiction,
facteur de normalisation

L'évidence $P(x_i) = \sum_{k=1}^K P(y_i = c_k) \prod_{l=1}^d P(x_{il}|y_i = c_k)$ est un facteur d'échelle qui dépend uniquement qui dépend uniquement de $x_{i1}, x_{i2}, \dots, x_{il}, \dots, x_{id}$.

Application à la classification

Nous pouvons faire des prédictions de y , pour toute nouvelle valeur de $x \in \mathbb{R}^d$, en utilisant la règle du maximum a posteriori (MAP) :

$$\begin{aligned}\hat{y} &= \operatorname{argmax}_{y \in Y} P(y|x) \\ &= \operatorname{argmax}_{y \in Y} P(y = c_k) \prod_{l=1}^d P(x_l | y = c_k)\end{aligned}$$

Estimation des paramètres du modèle Naïve bayes

➤ *Estimation de la probabilité a priori* $P(y_i = c_k)$
étant donnée une bases d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ avec $y_i \in \{c_1, c_2, \dots, c_k, \dots, c_K\}$

$$P(y_i = c_k) = \frac{n_k}{n}$$

n_k : nombre d'exemples (individus) appartenant à la classe c_k

n : nombre total d'exemples (individus).

Estimation des paramètres du modèle Naïve bayes

➤ *Estimation de la probabilité de la vraisemblance $P(x_{il}|y_i = c_k)$*

Cas discret ($x_{il} \in \{a_1, a_2, \dots, a_j, \dots, a_m\}$)

$$P(x_{il} = a_j | y_i = c_k) = \frac{n_{ljk}}{n_k}$$

n_{ljk} : nombre d'exemples (individus) ayant une valeur a_j , pour la variable l , appartenant à la classe c_k

n_k : nombre d'exemples (individus) appartenant à la classe c_k

Estimation des paramètres du modèle Naïve bayes

➤ *Estimation de la vraisemblance $P(x_{il}|y_i = c_k)$*

Cas continu ($x_{il} \in \mathbb{R}$)

Lorsqu'on traite des données continues, une hypothèse typique est que les valeurs continues associées à chaque classe sont distribuées selon une distribution normale (ou gaussienne).

$$P(x_{il}|y_i = c_k) = \frac{1}{\sigma_{kl}\sqrt{2\pi}} e^{-\frac{1}{2} \frac{(x_{il} - \mu_{kl})^2}{\sigma_{kl}^2}}$$

La moyenne μ_{kl} et l'écart type σ_{kl} peuvent être estimés à partir de la base d'apprentissage S .

Exemple : cas continu

petal.length	petal.width	variety
1.4	0.2	Setosa
1.4	0.2	Setosa
1.3	0.2	Setosa
1.5	0.2	Setosa
1.4	0.2	Setosa
5.8	1.6	Virginica
6.1	1.9	Virginica
6.4	2	Virginica
5.6	2.2	Virginica
5.1	1.5	Virginica

5.4	2.3	?
-----	-----	---

Exemple : cas discret

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Outlook	Temp	Humidity	Windy	Play
Rainy	Cool	High	True	?

Exemple : cas discret

Frequency Table

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3



Likelihood Table

		Play Golf	
		Yes	No
Outlook	Sunny	3/9	2/5
	Overcast	4/9	0/5
	Rainy	2/9	3/5

		Play Golf	
		Yes	No
Humidity	High		
	Normal		



		Play Golf	
		Yes	No
Humidity	High	.	.
	Normal	.	.

		Play Golf	
		Yes	No
Temp.	Hot		
	Mild	.	
	Cool		.



		Play Golf	
		Yes	No
Temp.	Hot	.	.
	Mild	.	.
	Cool		.

		Play Golf	
		Yes	No
Windy	False		
	True		.



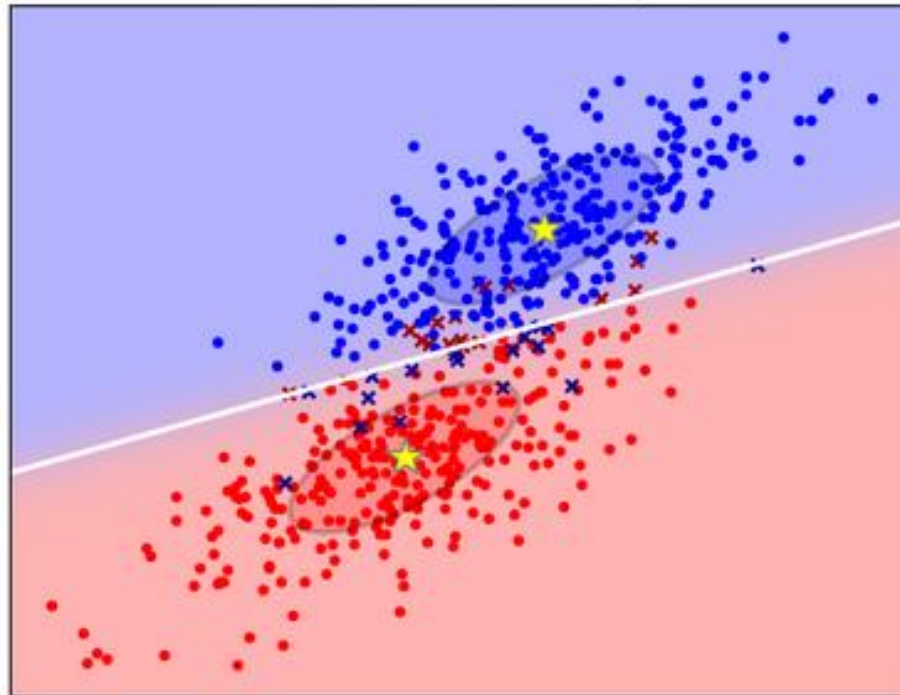
		Play Golf	
		Yes	No
Windy	False		.
	True	.	

Analyse discriminante linéaire/quadratique

- Introduction
- Apprentissage du modèle LDA
- Estimation des paramètres du modèle LDA
- Analyse discriminante quadratique
- Apprentissage du modèle QDA
- Estimation des paramètres du modèle QDA

Introduction

L'analyse discriminante linéaire (linear discriminant analysis (LDA)) est une méthode d'apprentissage supervisé qui fait partie des méthodes dites linéaires). Cette méthode cherche à trouver une frontière (linéaire) de discrimination entre les classes, qui peut s'écrire sous forme d'une ou plusieurs combinaisons linéaires des covariables.



Apprentissage du modèle LDA

Soit un ensemble d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ où la variable d'entrée x_i appartient à \mathbb{R}^d et la variable d'intérêt y_i appartient à un ensemble fini à K classes $Y = \{c_1, c_2, \dots, c_k, \dots, c_K\}$

$$P(y_i = c_k | x_i) = \frac{P(y_i = c_k)P(x_i | y_i = c_k)}{P(x_i)}$$

Apprentissage du modèle LDA: hypothèse de normalité

Supposons que x_i suit une loi normale multidimensionnelle (loi multi-normale) conditionnellement aux classes (y_i)

On peut écrire

$$P(x_i | y_i = c_k ; \theta_k) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{\det(\Sigma_k)}} e^{-\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)}$$

$\theta_k = (\mu_k, \Sigma_k)$: Paramètres du modèle

$\mu_k \in \mathbb{R}^d$: Vecteur des moyennes des x_i avec $y_i = c_k$

$\Sigma_k \in \mathbb{R}^{d \times d}$ Matrice de covariance des x_i avec $y_i = c_k$

Apprentissage du modèle LDA: hypothèse d'Homoscédasticité

Dans le cas de l'analyse discriminante linéaire on suppose toutes les classes ont la même matrice de covariance

$$\Sigma = \Sigma_1 = \Sigma_2 = \dots = \Sigma_k = \dots = \Sigma_K$$

Apprentissage du modèle LDA: règle de prédiction

Pour toute nouvelle valeur de $x \in \mathbb{R}^d$, la règle de prédiction de y peut se faire par le maximum a posteriori (MAP) :

$$\hat{y} = \operatorname{argmax}_{k \in \{1, 2, \dots, K\}} P(y=c_k|x; \theta_k)$$

$$\begin{aligned} \text{Avec } P(y=c_k|x; \theta_k) &= \frac{P(y=c_k)P(x|y=c_k;\theta_k)}{P(x)} = \frac{P(y=c_k)P(x|y=c_k;\theta_k)}{\sum_{g=1}^K P(y=c_g)P(x|y=c_g;\theta_g)} \\ &= \frac{P(y=c_k)\mathcal{N}(x,\mu_k,\Sigma)}{\sum_{g=1}^K P(y=c_g)\mathcal{N}(x,\mu_g,\Sigma)} \end{aligned}$$

Apprentissage du modèle LDA: frontière de décision

La frontière (limite) de décision entre les classes $= c_k$ et $= c_g$, pour un ensemble de données x , vérifie l'égalité suivante:

$$P(y=c_k|x; \theta_k) = P(y=c_g|x; \theta_g)$$

Ainsi

$$\frac{P(y=c_k|x; \theta_k)}{P(y=c_g|x; \theta_g)} = 1 \Leftrightarrow \log \frac{P(y=c_k|x; \theta_k)}{P(y=c_g|x; \theta_g)} = 0$$

$$\begin{aligned} \log \frac{P(y=c_k|x; \theta_k)}{P(y=c_g|x; \theta_g)} &= \log \frac{P(y=c_k)}{P(y=c_g)} + \log \frac{\mathcal{N}(x, \mu_k, \Sigma)}{\mathcal{N}(x, \mu_g, \Sigma)} \\ &= \log \frac{P(y=c_k)}{P(y=c_g)} + -\frac{1}{2} [\mu_k^T \Sigma^{-1} \mu_k - \mu_g^T \Sigma^{-1} \mu_g + 2x^T \Sigma^{-1} (\mu_k - \mu_g)] \end{aligned}$$

Cette fonction est linéaire par rapport à x , on obtient alors des fonctions discriminantes linéaires.

Estimation des paramètres du modèle LDA

➤ *Estimation de la probabilité a priori $P(y_i = c_k)$*

Etant donnée une bases d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ avec $y_i \in \{c_1, c_2, \dots, c_k, \dots, c_K\}$

Les probabilités a priori $P(y=c_k)$ de chaque classe est calculée en se basant sur la proportion de la classe c_k dans l'ensemble des données de d'apprentissage:

$$P(y=c_k) = \frac{n_k}{n}$$

n_k : nombre d'exemples (individus) apparentant à la classe c_k

n : nombre total d'exemples (individus).

Estimation des paramètres du modèle LDA

➤ *Estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$*

L'estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$ revient à estimer $\theta_k = (\mu_k, \Sigma_k)$

Maximisation du log-vraisemblance $\mathcal{L}(\theta_k; \mathbf{x})$, donnée par:

$$\begin{aligned}\mathcal{L}(\theta_k; \mathbf{x}) &= \text{Log} \prod_{i|y_i=c_k} P(y_i = c_k) \mathcal{N}(x_i, \mu_k, \Sigma) \\ &= \text{Log}(P(y = c_k)) + \sum_{i|y_i=c_k} \log(\mathcal{N}(x_i, \mu_k, \Sigma))\end{aligned}$$

Estimation des paramètres du modèle LDA

➤ *Estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$*

les paramètres μ_k et Σ_k peuvent être obtenus par la résolution des équations:

$$\frac{\delta \mathcal{L}(\theta_k; \mathbf{x})}{\delta \mu_k} = 0$$

et

$$\frac{\delta \mathcal{L}(\theta_k; \mathbf{x})}{\delta \Sigma_k} = 0$$

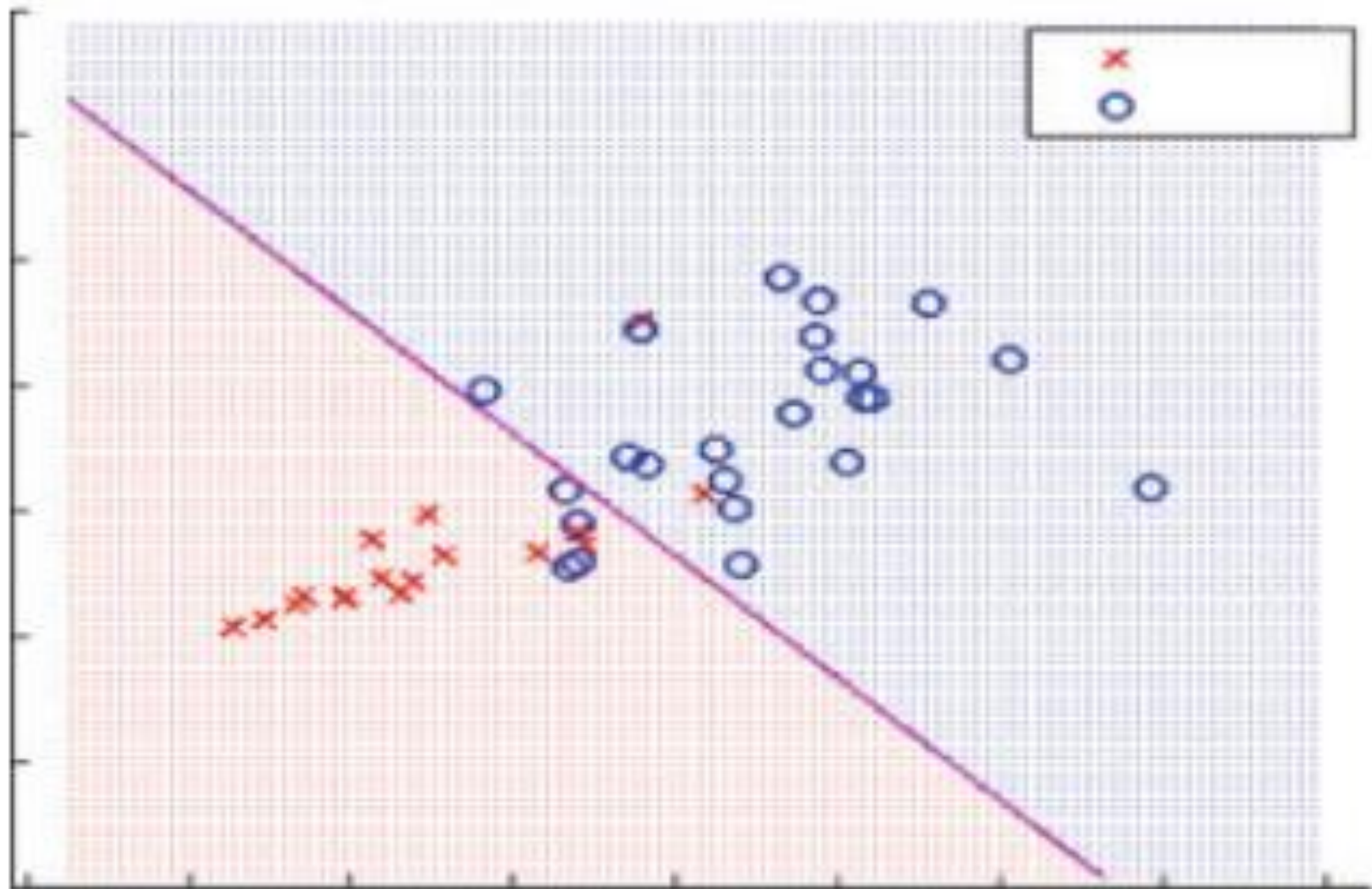
Estimation des paramètres du modèle LDA

➤ *Estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$*

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i|y_i=c_k} x_i$$

$$\hat{\Sigma} = \hat{\Sigma}_k = \frac{1}{n - K} \sum_{k=1}^K \sum_{i|y_i=c_k} (x_i - \hat{\mu}_k)^T (x_i - \hat{\mu}_k)$$

Illustration : LDA



L'analyse discriminante quadratique

L'analyse discriminante quadratique, en anglais Quadratic Discriminant Analysis (QDA), est étroitement liée à l'analyse discriminante linéaire (LDA), où l'on suppose que les mesures de chaque classe sont normalement distribuées (**hypothèse de normalité**). Cependant, contrairement à la LDA, la QDA ne suppose pas que la covariance de chacune des classes soit identique (**hypothèse d'hétéroscédasticité**) .

$$\Sigma_1 \neq \Sigma_2 \dots \neq \Sigma_k \neq \dots \neq \Sigma_K$$

Apprentissage du modèle QDA: frontière de décision

La frontière de décision entre les classes c_k et c_g

$$\begin{aligned} \log \frac{P(y=c_k|x; \theta_k)}{P(y=c_g|x; \theta_g)} &= \log \frac{P(y=c_k)}{P(y=c_g)} + \log \frac{\mathcal{N}(x, \mu_k, \Sigma_k)}{\mathcal{N}(x, \mu_g, \Sigma_g)} \\ &= \log \frac{P(y=c_k)}{P(y=c_g)} + \frac{1}{2} \log \left(\frac{\Sigma_k}{\Sigma_g} \right) - \frac{1}{2} [(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \\ &\quad (x - \mu_g)^T \Sigma_g^{-1} (x - \mu_g)] \end{aligned}$$

Cette fonction est quadratique par rapport à x , on obtient alors des fonctions discriminantes quadratiques.

Estimation des paramètres du modèle QDA

➤ *Estimation de la probabilité a priori $P(y_i = c_k)$*

$$P(y=c_k) = \frac{n_k}{n}$$

n_k : nombre d'exemples (individus) appartenant à la classe c_k

n : nombre total d'exemples (individus).

Estimation des paramètres du modèle QDA

➤ *Estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$*

Les paramètres de la QDA sont estimés de la même manière que pour la LDA, mis à part les matrices de covariance qui doivent être estimées pour chaque classe.

Maximisation du log-vraisemblance $\mathcal{L}(\theta_k; \mathbf{x})$, donnée par:

$$\begin{aligned}\mathcal{L}(\theta_k; \mathbf{x}) &= \text{Log} \prod_{i|y_i=c_k} P(y_i = c_k) \mathcal{N}(x_i, \mu_k, \Sigma_k) \\ &= \text{Log}(P(y = c_k)) + \sum_{i|y_i=c_k} \log(\mathcal{N}(x_i, \mu_k, \Sigma_k))\end{aligned}$$

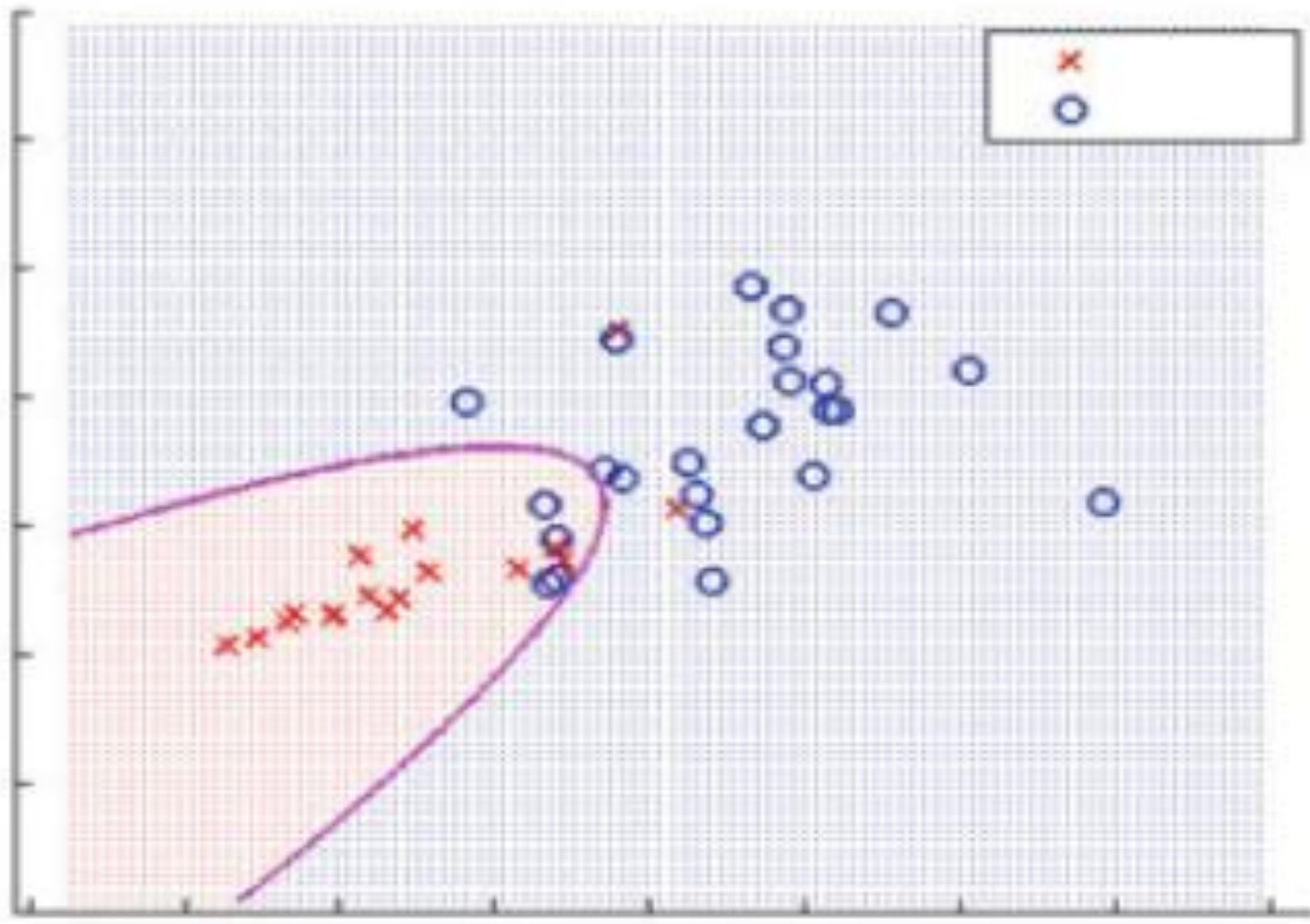
Estimation des paramètres du modèle QDA

➤ *Estimation de la vraisemblance $P(x|y = c_k ; \theta_k)$*

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i|y_i=c_k} x_i$$

$$\hat{\Sigma}_k = \frac{1}{n_k - 1} \sum_{i|y_i=c_k} (x_i - \hat{\mu}_k)^T (x_i - \hat{\mu}_k)$$

Illustration : QDA



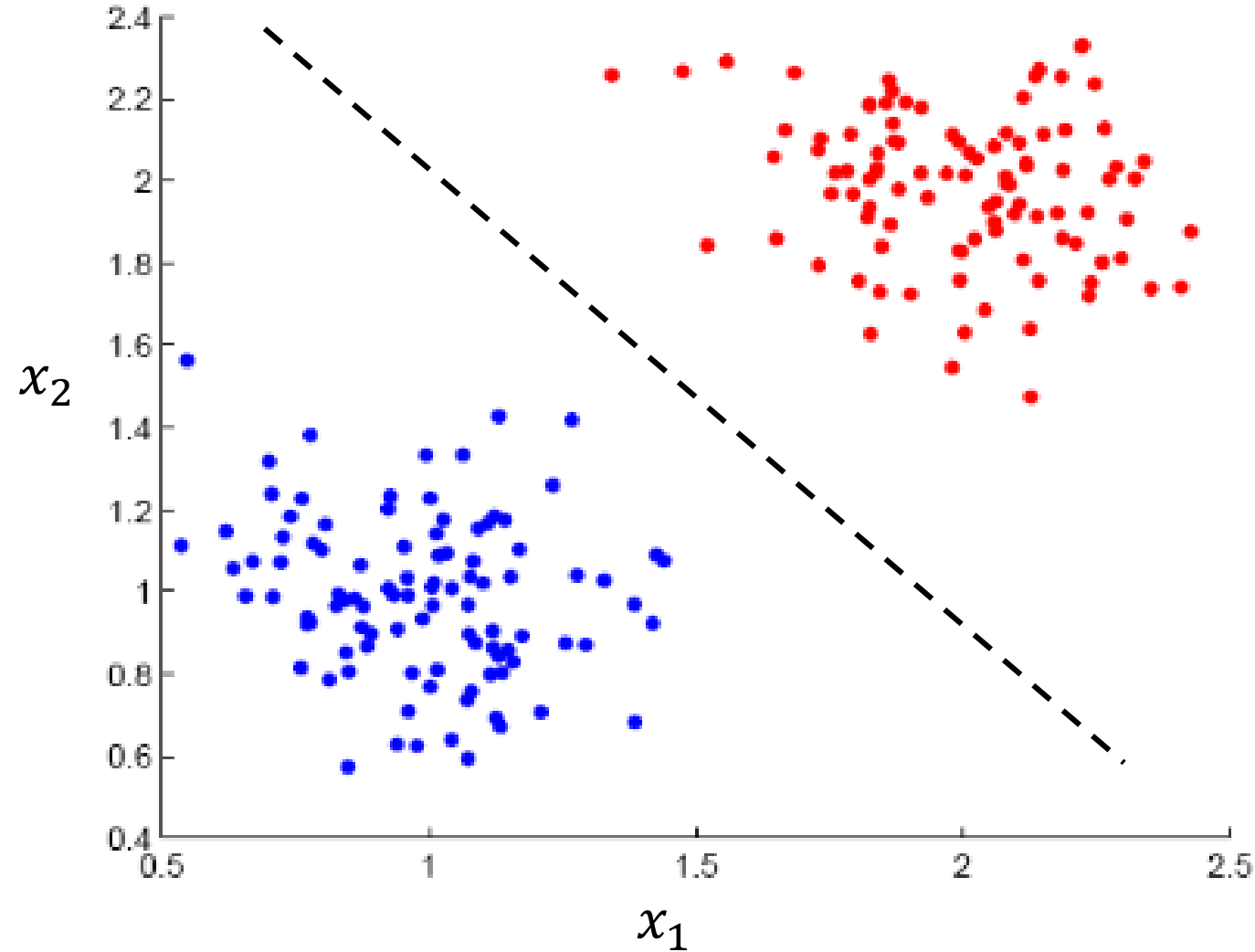
Séparateurs à Vaste Marge

- Introduction
- Principe
- Cas bi-classes
 - Données linéairement séparables
 - Données non séparables (marge souple)
 - Données non linéairement séparables (Astuce noyau)
- Cas multi-classes
 - One vs all
 - One vs one

Introduction

- Le modèle des Séparateurs à Vaste Marge (support vector machine SVM) est un modèle d'apprentissage supervisé introduit par Vapnik pour résoudre les problèmes de la classification et de la régression.
- Initialement développé pour la classification binaire.
- Les SVM reposent sur deux notions clés :
 - La notion de marge maximale
 - La notion de fonction noyau

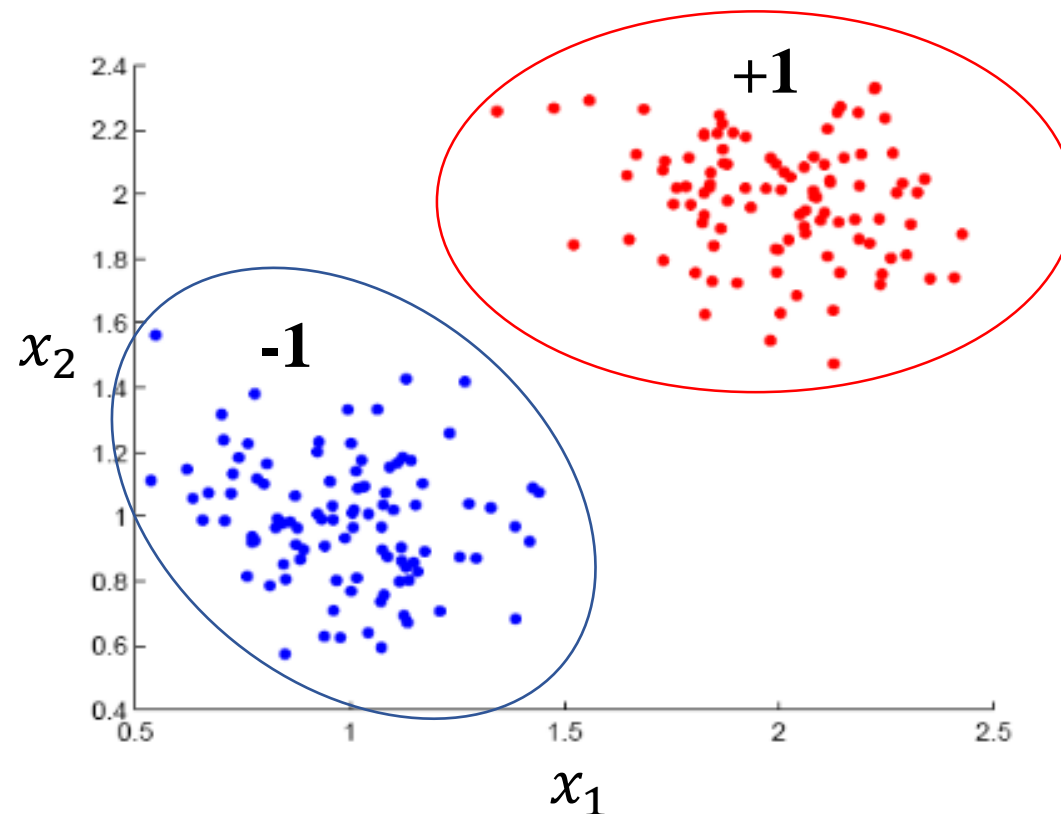
Principe



Le principe des SVM repose sur l'idée de recherche d'une règle de décision basée sur une séparation par hyperplan de marge maximale.

Principe : cas bi-classes

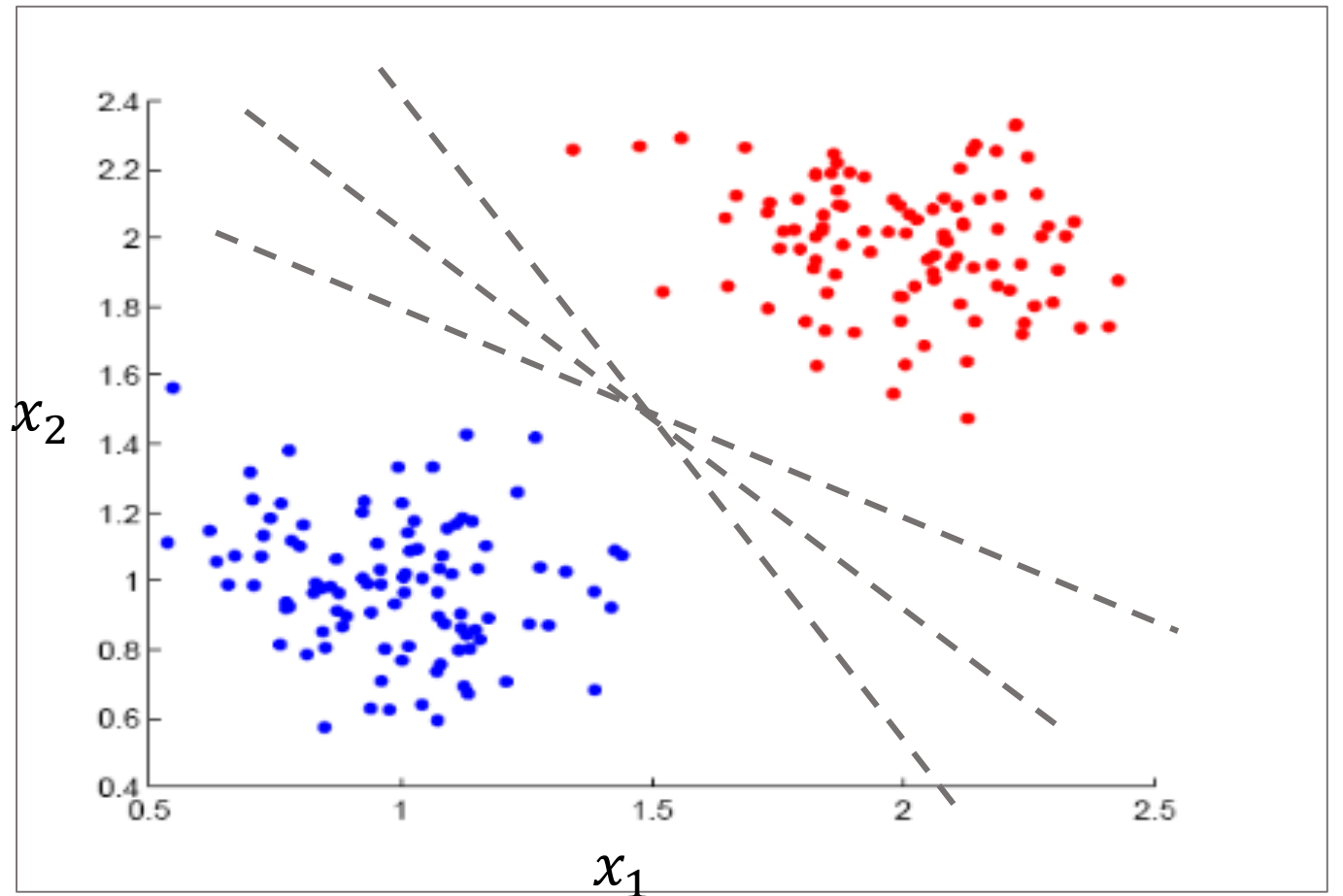
Soit un ensemble d'apprentissage $S = \{(x_i, y_i)\}_{i \leq n}$ où la variable d'entrée x_i appartient à \mathbb{R}^d et la variable d'intérêt y_i appartient à un ensemble fini à 2 classes, $Y = \{+1, -1\}$



Principe : cas bi-classes

Cas de données linéairement séparables: trouver une fonction de prédiction linéaire

$$w^T x + b = 0$$



Principe : cas bi-classes

Cas de données linéairement séparables: règle de décision

Projection de \vec{x} sur \vec{w}

$$\vec{w} \cdot \vec{x} \geq c$$

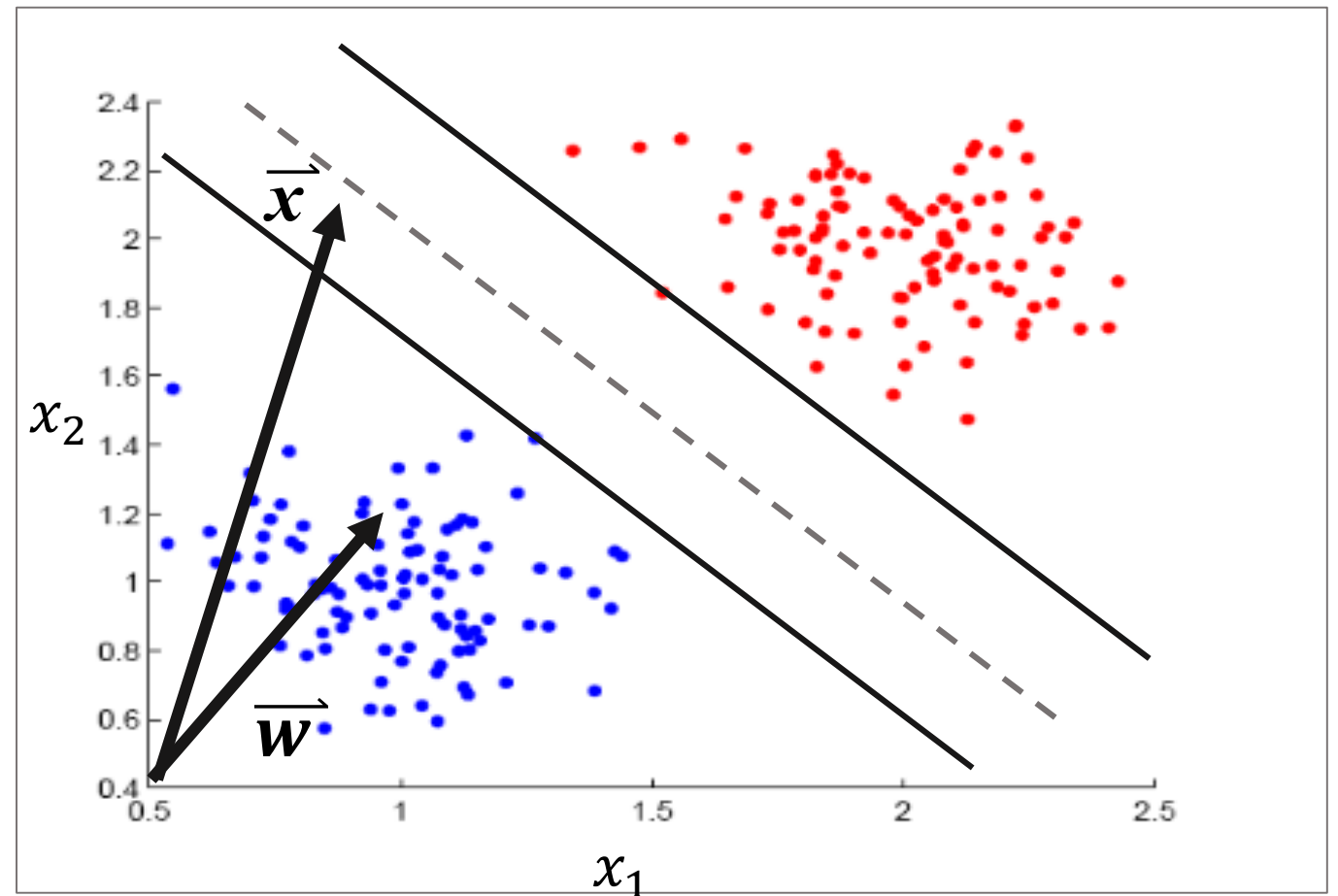
$$\vec{w} \cdot \vec{x} - c \geq 0$$

$$\vec{w} \cdot \vec{x} + b \geq 0 \Rightarrow \mathbf{x \text{ est } +}$$

$$\vec{w} \cdot \vec{x} + b \leq 0 \Rightarrow \mathbf{x \text{ est } -}$$

Avec

$$b = -c$$



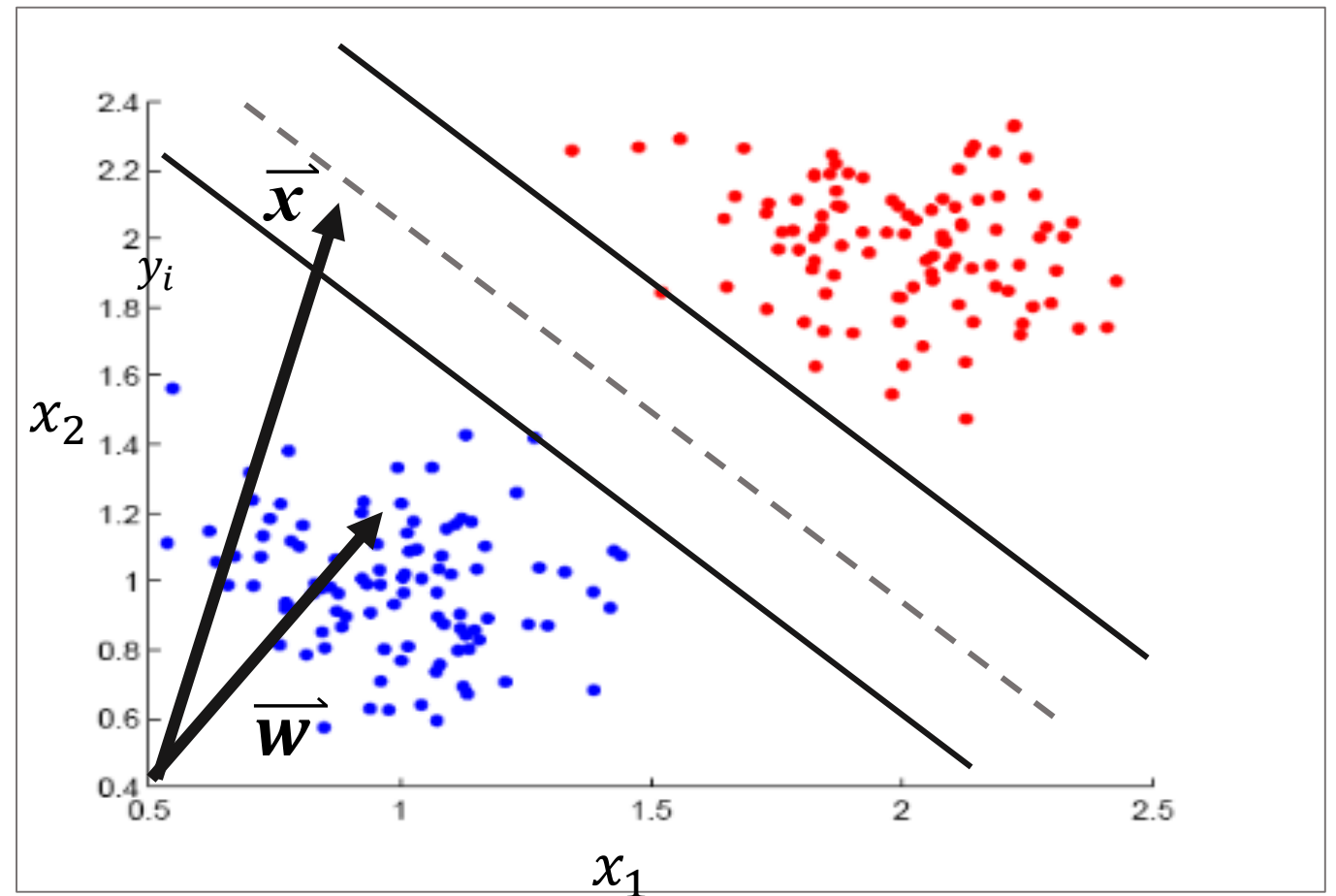
Principe : cas bi-classes

Cas de données linéairement séparables: règle de décision

La frontière de décision

$$f(\mathbf{x}) = \overrightarrow{w} \cdot \overrightarrow{x} + b = 0$$

1



Principe : cas bi-classes

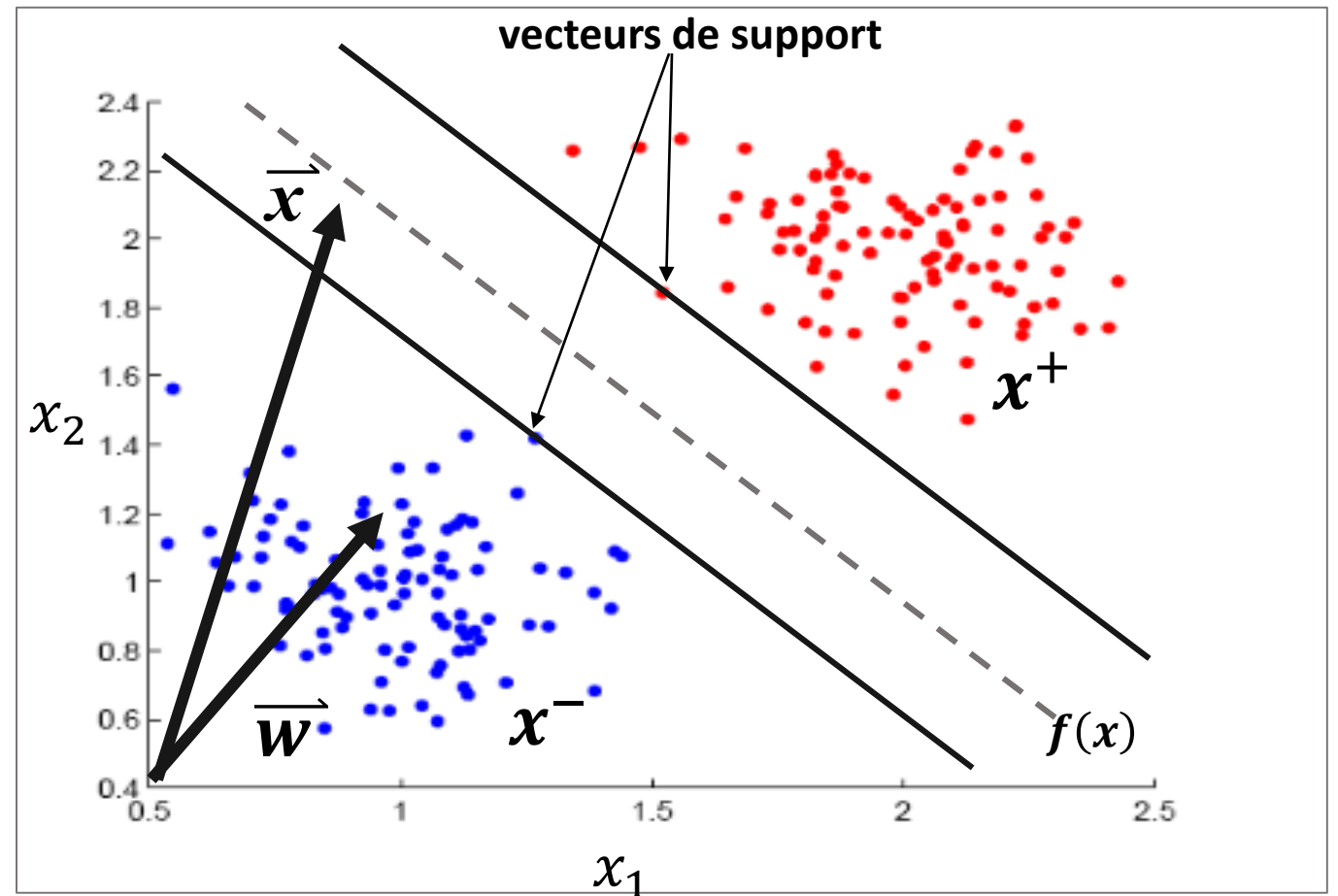
Cas de données linéairement séparables: estimation de de la fonction de perdition $f(x)$

$$f(x) = \overrightarrow{w} \cdot \overrightarrow{x} + b = 0$$

Avec

$$\overrightarrow{w} \cdot \overrightarrow{x^+} + b \geq 1$$

$$\overrightarrow{w} \cdot \overrightarrow{x^-} + b \leq -1$$



Principe : cas bi-classes

Cas de données linéairement séparables: estimation de la fonction de perte $f(x)$

Pour tout x_i - on a $y_i = +1 \Rightarrow y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1$

Pour tout x_i - on a $y_i = -1 \Rightarrow y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1$

$\forall (x_i, y_i)$ on a $y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1$ 2

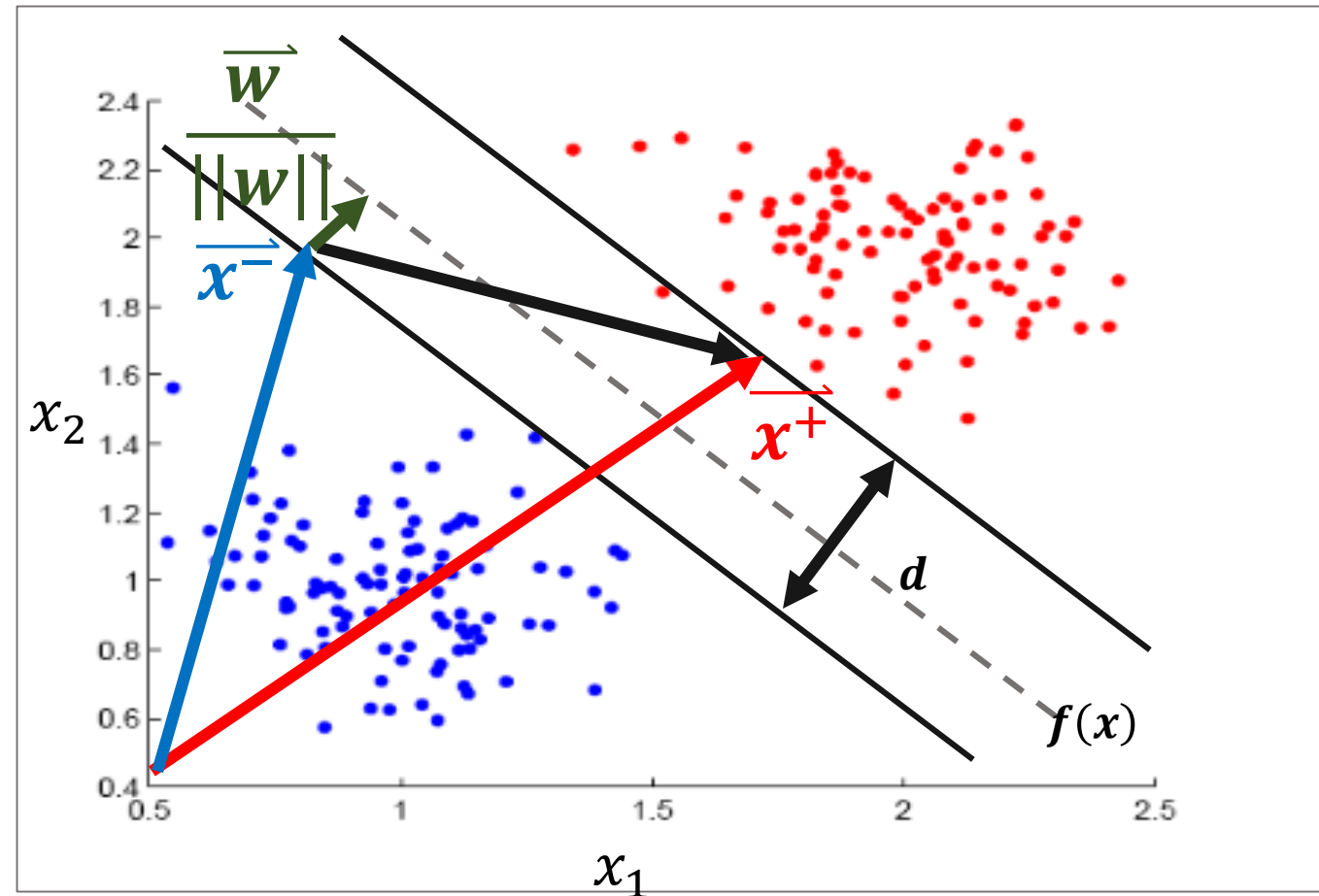
Principe : cas bi-classes

Cas de données linéairement séparables: estimation de de la fonction de perdition $f(x)$

Maximiser la distance d

$$d = (\overrightarrow{x^+} - \overrightarrow{x^-}) \cdot \frac{\overrightarrow{w}}{\|w\|} = \frac{2}{\|w\|}$$

Maximiser $\frac{2}{\|w\|} \Rightarrow$ Minimiser $\|w\|$
 \Rightarrow Minimiser $\frac{\|w\|^2}{2}$



Principe : cas bi-classes

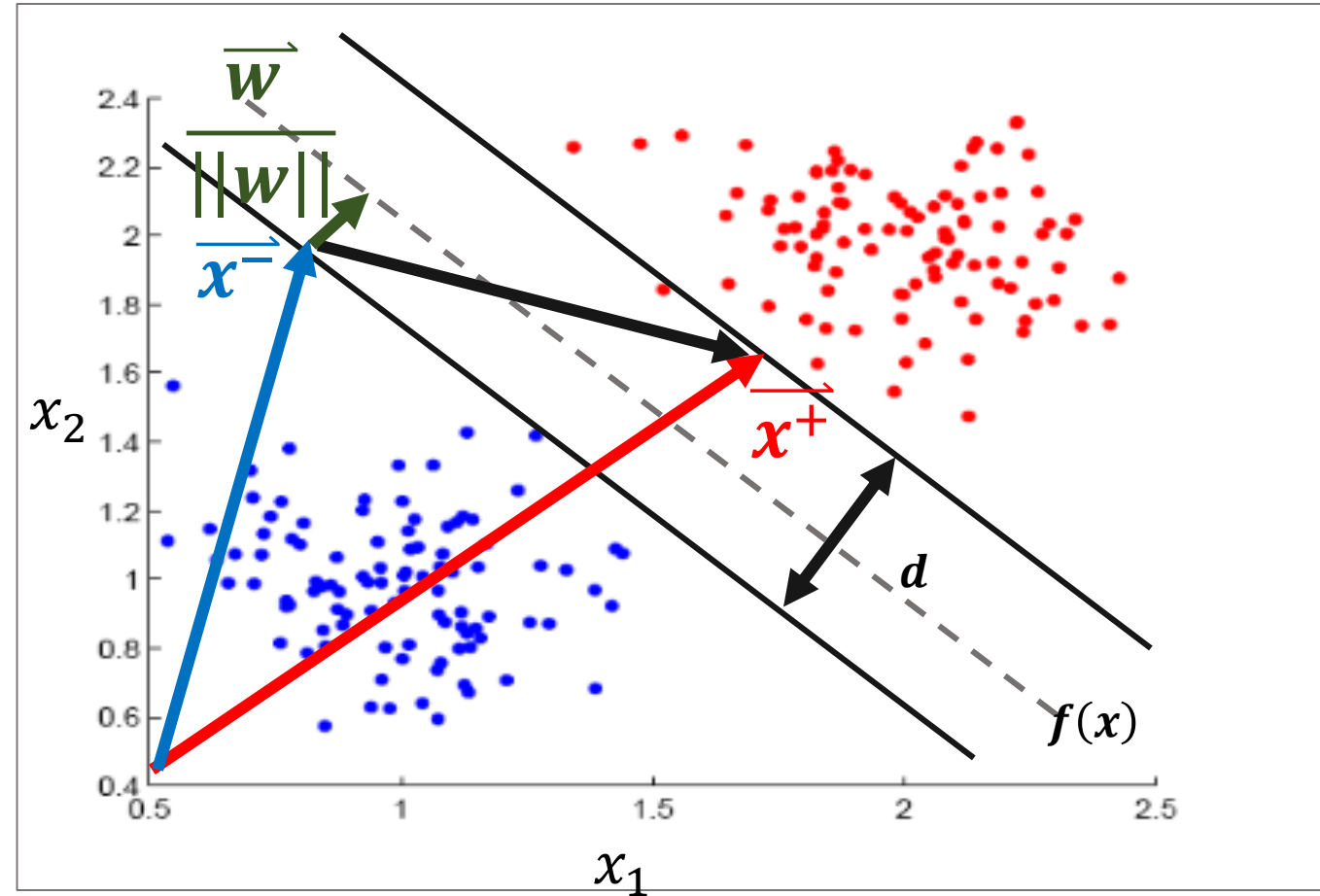
Cas de données linéairement séparables: estimation de de la fonction de perdition

$f(x)$

➤ P. Primal

$$\begin{cases} \text{Minimiser } \frac{||\mathbf{w}||^2}{2} \\ \text{s. c} \\ y_i(\overline{\mathbf{w}} \cdot \overline{\mathbf{x}}_i + b) \geq 1 \end{cases}$$

Ce problème d'optimisation peut être résolu par la méthode du Lagrangien



Principe : cas binaire (deux classes)

Cas de données linéairement séparables: estimation de la fonction de perte $f(x)$

Conditions de Karush – Kuhn – Tucker

$$\mathcal{L}(w, b, \alpha) = \frac{\|\mathbf{w}\|^2}{2} - \sum_{i=1}^n \alpha_i (y_i (\overrightarrow{\mathbf{w}} \cdot \overrightarrow{x_i} + b) - 1)$$

$$\frac{\delta \mathcal{L}(w, b, \alpha)}{\delta \mathbf{w}} = 0 \Rightarrow \mathbf{w} - \sum_{i=1}^n \alpha_i y_i \cdot \overrightarrow{x_i} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^n \alpha_i y_i \cdot \overrightarrow{x_i}$$

$$\frac{\delta \mathcal{L}(w, b, \alpha)}{\delta b} = 0 \Rightarrow - \sum_{i=1}^n \alpha_i y_i = 0 \Rightarrow \sum_{i=1}^n \alpha_i y_i = 0$$

$\alpha_i \geq 0$ multiplicateurs de Lagrange

Principe : cas bi-classes

Cas de données linéairement séparables: estimation de la fonction de perte $f(\mathbf{x})$

➤ **P. dual**

$$\begin{aligned}\mathcal{L}(\alpha) &= \frac{1}{2} \left(\sum_{i=1}^n \alpha_i y_i \cdot \vec{x_i} \sum_{j=1}^n \alpha_j y_j \cdot \vec{x_j} \right) - \left(\sum_{i=1}^n \alpha_i y_i \vec{x_i} \sum_{j=1}^n \alpha_j y_j \cdot \vec{x_j} \right) - \sum_{j=1}^n \alpha_j y_j b + \sum_{i=1}^n \alpha_i \\ &= -\frac{1}{2} \sum_{i=1}^n \alpha_i y_i \vec{x_i} \sum_{j=1}^n \alpha_j y_j \cdot \vec{x_j} + \sum_{i=1}^n \alpha_i \\ &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \vec{x_i} \cdot \vec{x_j} + \sum_{i=1}^n \alpha_i\end{aligned}$$

Principe : cas bi-classes

Cas de données linéairement séparables: estimation de la fonction de perte $f(\mathbf{x})$

➤ **P. dual**

$$\left\{ \begin{array}{l} \text{Maximiser } \mathcal{L}(\alpha) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \overrightarrow{x_i} \cdot \overrightarrow{x_j} + \sum_{i=1}^n \alpha_i \\ \text{S. C} \\ \alpha_i \geq 0 \\ \sum_{i=1}^n \alpha_i y_i = \mathbf{0} \end{array} \right.$$

Principe : cas bi-classes

Cas de données linéairement séparables: estimation de la fonction de perte $f(\mathbf{x})$

On cherche α^* qui maximise $\mathcal{L}(\alpha)$ (optimisation quadratique)

$$\mathbf{w}^* = \sum_{i=1}^n \alpha_i^* y^i \cdot \overrightarrow{x_i}$$

$$\mathbf{b}^* = \frac{1}{y_s} - \overrightarrow{\mathbf{w}^*} \cdot \overrightarrow{x_s} \text{ avec } x_s \text{ represent les vecteurs de support } \alpha_i^* > 0$$

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y^i \cdot \overrightarrow{x_i} \cdot \overrightarrow{x} + \frac{1}{y_s} - \sum_{i=1}^n \alpha_i^* y^i \cdot \overrightarrow{x_i} \cdot \overrightarrow{x_s}$$

Principe : cas bi-classes

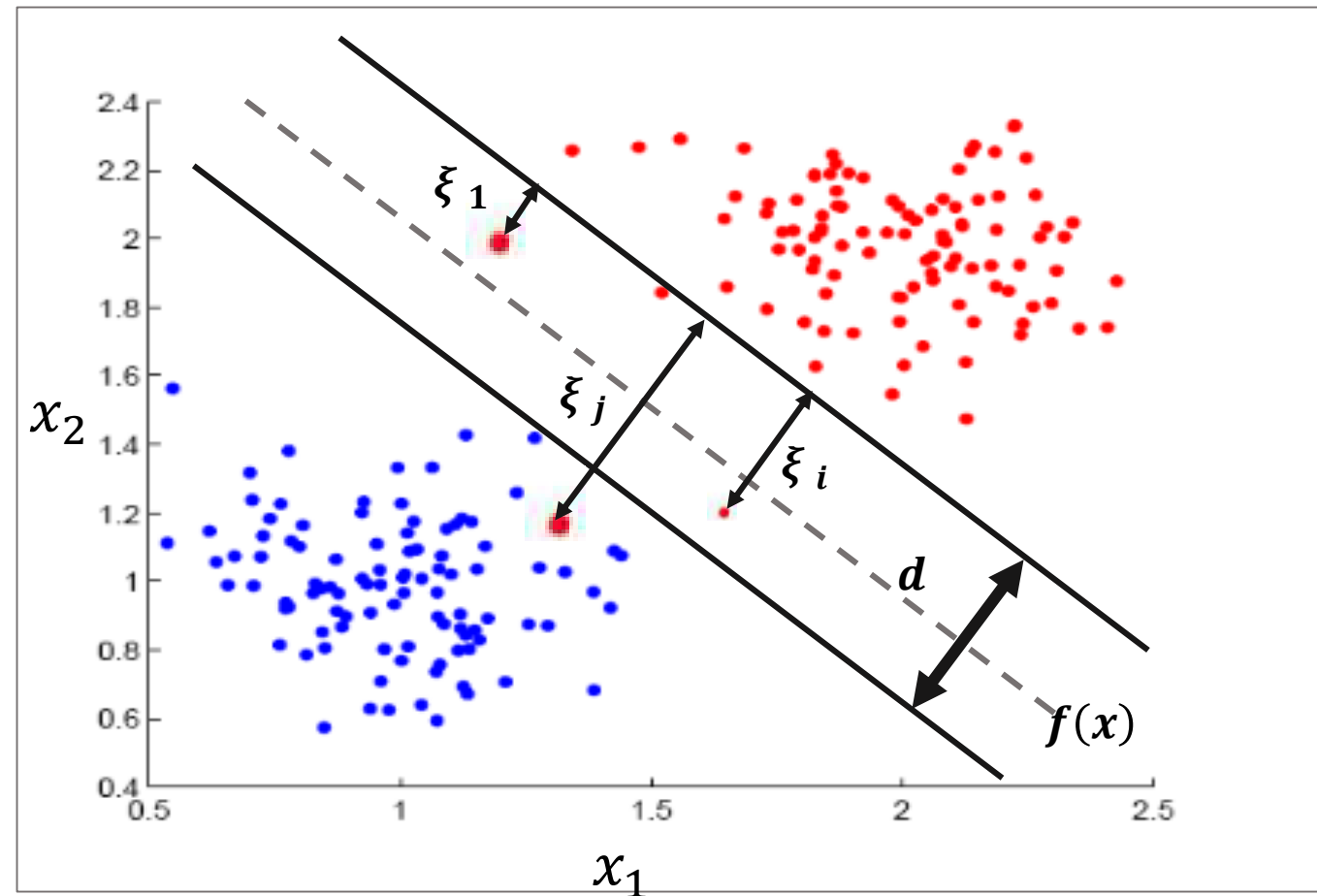
Cas de données non séparables: marge souple

➤ P. Primal

$$\left\{ \begin{array}{l} \text{Minimiser } \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^n \xi_i \\ \text{s. c} \\ y_i(\overline{\mathbf{w}} \cdot \overline{x_i} + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{array} \right.$$

ξ_i : variable ressort

$C > 0$: constante de tolérance à ajuster



Principe : cas bi-classes

Cas de données non séparables: marge souple

➤ P. dual

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \lambda) = \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i (y_i (\overline{\mathbf{w}} \cdot \overline{x_i} + b) + 1 - \xi_i) - \sum_{i=1}^n \lambda_i \xi_i$$

Principe : cas bi-classes

Cas de données non séparables: marge souple

Conditions de Karush – Kuhn – Tucker

$$\frac{\delta \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \alpha)}{\delta \mathbf{w}} = 0 \Rightarrow \mathbf{w} - \sum_{i=1}^n \alpha_i y_i \cdot \overrightarrow{x_i} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^n \alpha_i y_i \cdot \overrightarrow{x_i}$$

$$\frac{\delta \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \alpha)}{\delta b} = 0 \Rightarrow - \sum_{i=1}^n \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_{i=1}^n \alpha_i y_i = 0$$

$$\frac{\delta \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \alpha)}{\delta \xi} = 0 \Rightarrow \mathbf{C} - \alpha_i - \lambda_i = 0 \quad \Rightarrow \quad \lambda_i = \alpha_i - \mathbf{C}$$

$$\lambda_i \geq \mathbf{0}$$

$$\alpha_i \leq \mathbf{C}$$

Principe : cas bi-classes

Cas de données non séparables: marge souple

➤ P. dual

Par substitution on obtient

$$\mathcal{L}(\alpha) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \overrightarrow{x_i} \cdot \overrightarrow{x_j} + \sum_{i=1}^n \alpha_i$$

Principe : cas bi-classes

Cas de données non séparables: marge souple

➤ P. dual

$$\left\{ \begin{array}{l} \text{Maximiser } \mathcal{L}(\alpha) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \overrightarrow{x_i} \cdot \overrightarrow{x_j} + \sum_{i=1}^n \alpha_i \\ \text{S. C} \\ 0 \leq \alpha_i \leq C \\ \sum_{i=1}^n \alpha_i y_i = \mathbf{0} \end{array} \right.$$

Principe : cas bi-classes

Cas de données non séparables: marge souple

On cherche α^* qui maximise $\mathcal{L}(\alpha)$ (optimisation quadratique)

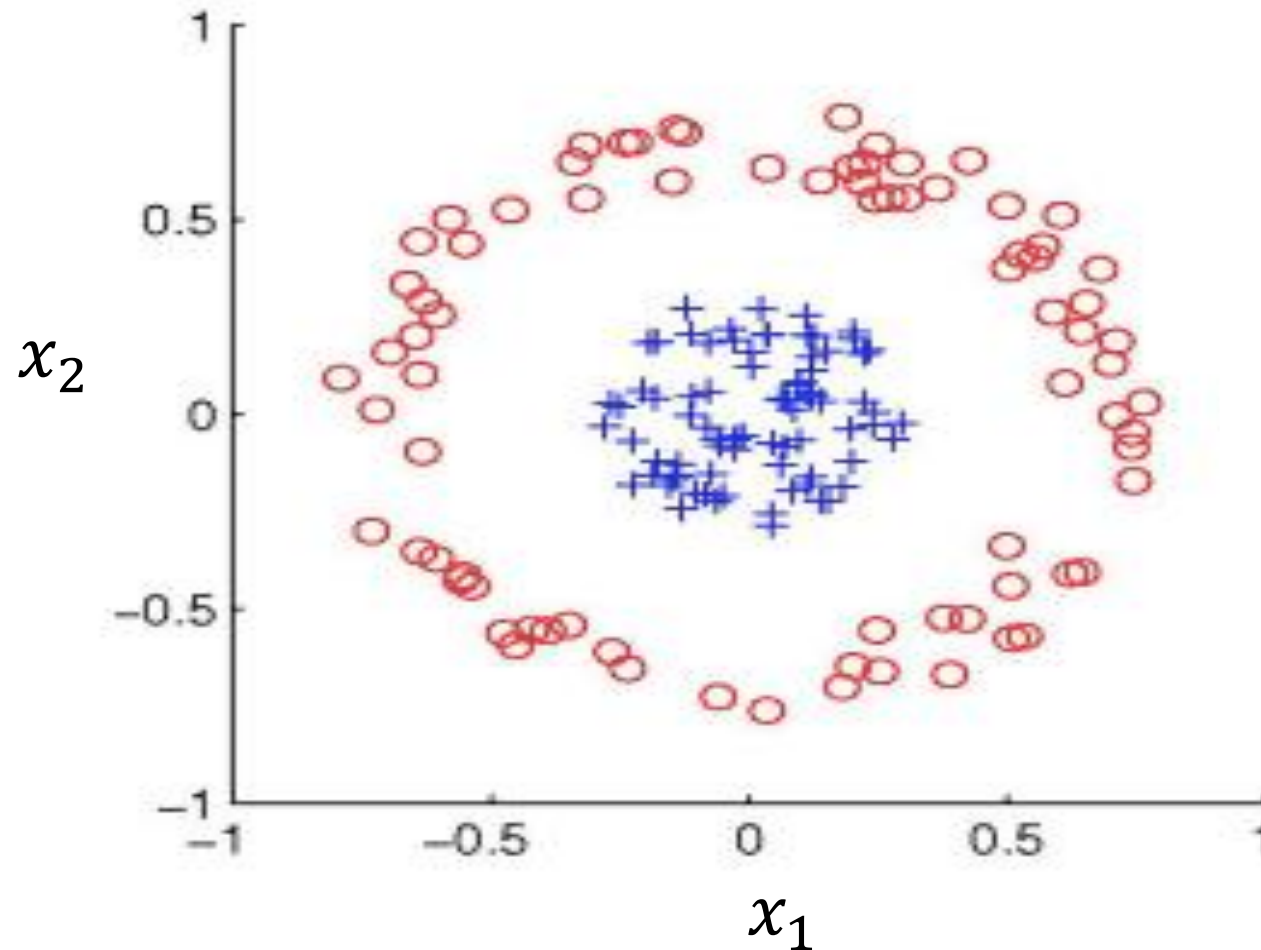
$$\mathbf{w}^* = \sum_{i=1}^n \alpha_i^* y^i . \overrightarrow{x_i}$$

$$\mathbf{b}^* = \frac{1}{y_s} - \overrightarrow{\mathbf{w}^*} . \overrightarrow{x_s} \text{ pour } 0 < \alpha_i < C$$

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y^i . \overrightarrow{x_i} . \overrightarrow{\mathbf{x}} + \mathbf{b}^*$$

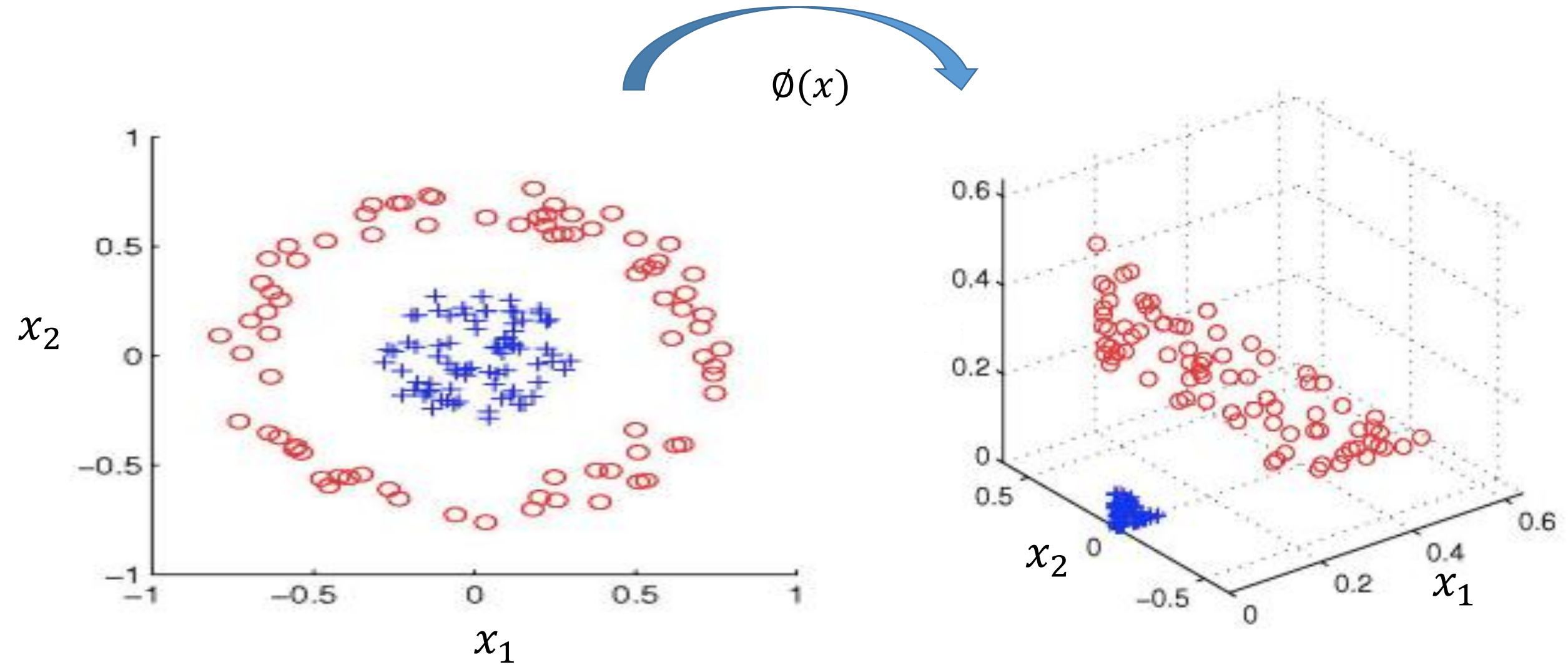
Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)



Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)



Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

$\phi(x)$: est fonction de projection

$$\phi : \mathcal{X} \rightarrow \mathcal{H}$$

- \mathcal{F} est un sous ensemble d'un espace vectoriel (ou d'Hilbert) de dimension supérieure à \mathcal{X} , $\text{card}(\mathcal{H}) > \text{card}(\mathcal{X})$
- \mathcal{H} est l'espace de redescription.

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

On obtient un l'ensemble $S' \triangleq \{(\phi(x_i), y_i)\}_{i \leq n}$ où la variable d'entrée $\phi(x_i) \in \mathcal{F}$ et la variable d'intérêt y_i appartient à un ensemble fini à 2 classes, $Y = \{+1, -1\}$

La fonction de prédiction peut s'écrire :

$$f(\mathbf{x}) = \overrightarrow{w} \cdot \overrightarrow{\phi(\mathbf{x})} + b$$

Problème : comment choisir ϕ

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y_i \cdot \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)} + \frac{1}{y_s} - \sum_{i=1}^n \alpha_i^* y_i \cdot \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x_s)}$$

Problème : comment choisir ϕ

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y_i \cdot \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)} + \frac{1}{y_s} - \sum_{i=1}^n \alpha_i^* y_i \cdot \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x_s)}$$

- Pb: Le produit scalaire $\overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)}$ est effectué dans un espace de grande dimension, ce qui conduit à des calculs impraticables.
- Remplacer ce calcul $(\overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)})$ par une fonction noyau K :

$$K(x_i, x) = \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)}$$

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

Théorème de Mercer

Pour tout noyau K positif sur \mathcal{X} , il existe un espace d'Hilbert \mathcal{H} et une application ϕ tels que:

$$K(x_i, x) = \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)}$$

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

Une fonction noyau est une fonction $K: x, x' \in \mathcal{X}^2 \rightarrow \mathbb{R}$

$$K(x, x') = \overrightarrow{\phi(x_i)} \cdot \overrightarrow{\phi(x)}$$

- $K(x, x')$ est une fonction noyau si et seulement si elle est symétrique et positive semi définie
- La fonction noyau permet de calculer implicitement un produit scalaire dans un espace de dimension élevé par un calcul n'impliquant qu'un nombre fini de termes

Principe : cas bi-classes

Cas de données non linéairement séparables: astuce du noyau (kernel trick)

Ainsi

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y_i \cdot K(x_i, x) + \frac{1}{y_s} - \sum_{i=1}^n \alpha_i^* y_i \cdot K(x_i, x)$$

Principe : cas bi-classes

Cas de données non linéairement séparables: Quelques fonctions noyaux

Noyau linéaire $K(x_i, x) = \vec{x}_i \cdot \vec{x}$

Noyau polynomiale $K(x_i, x) = (\vec{x}_i \cdot \vec{x})^p$, avec $p > 0$

Noyau gaussien $K(x_i, x) = e^{-\frac{\|x_i - x\|^2}{\sigma}}$

Cas multi-classes



- Transformer le problème de classification multi-classe un problème de classification bi-classes

Cas multi-classes

Deux stratégies

➤ One vs all

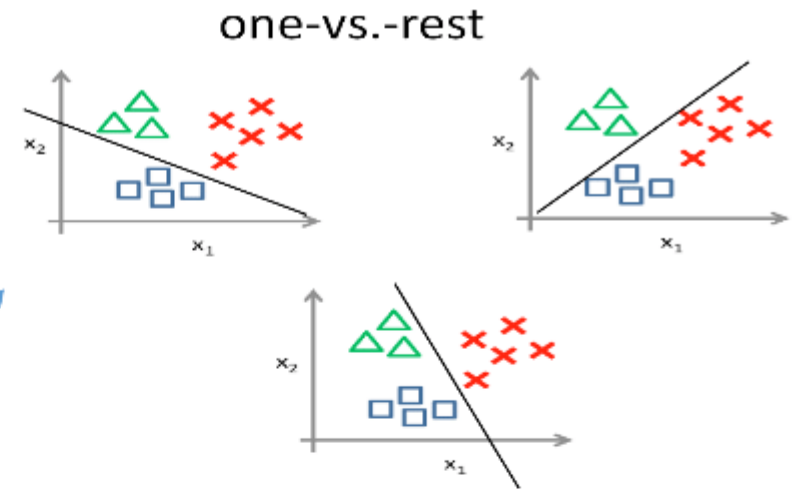
- (K fonctions discriminantes)

➤ One vs One

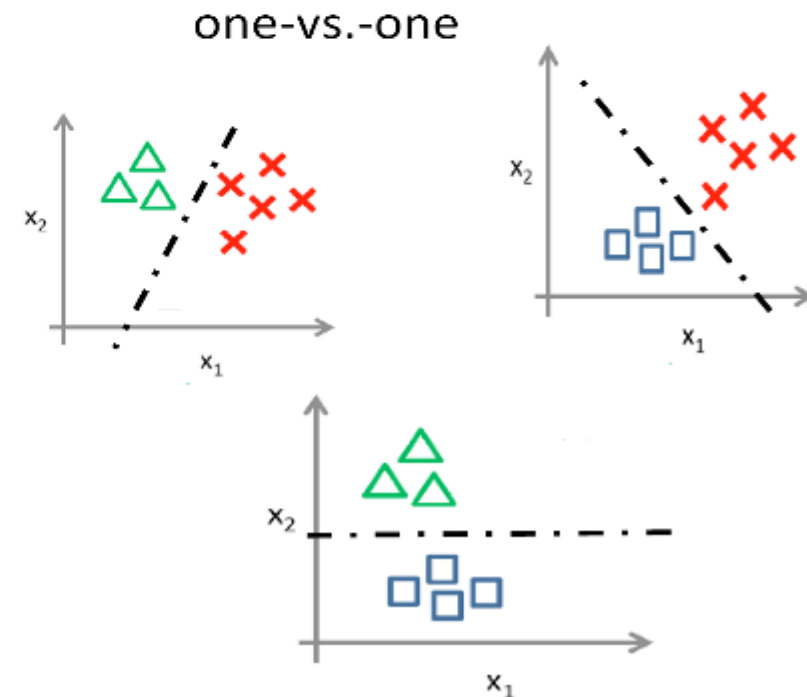
- $\frac{K(K-1)}{2}$ fonctions discriminantes



First strategy



second strategy



Cas multi-classes

Règle de décision

- **One vs all** : Choisir la classe prédite avec le meilleur score
- **One vs One** : Choisir la classe qui remporte le plus de duels.