

Cours 1

Catherine ACHARD
Institut des Systèmes Intelligents et de Robotique

catherine.achard@sorbonne-universite.fr

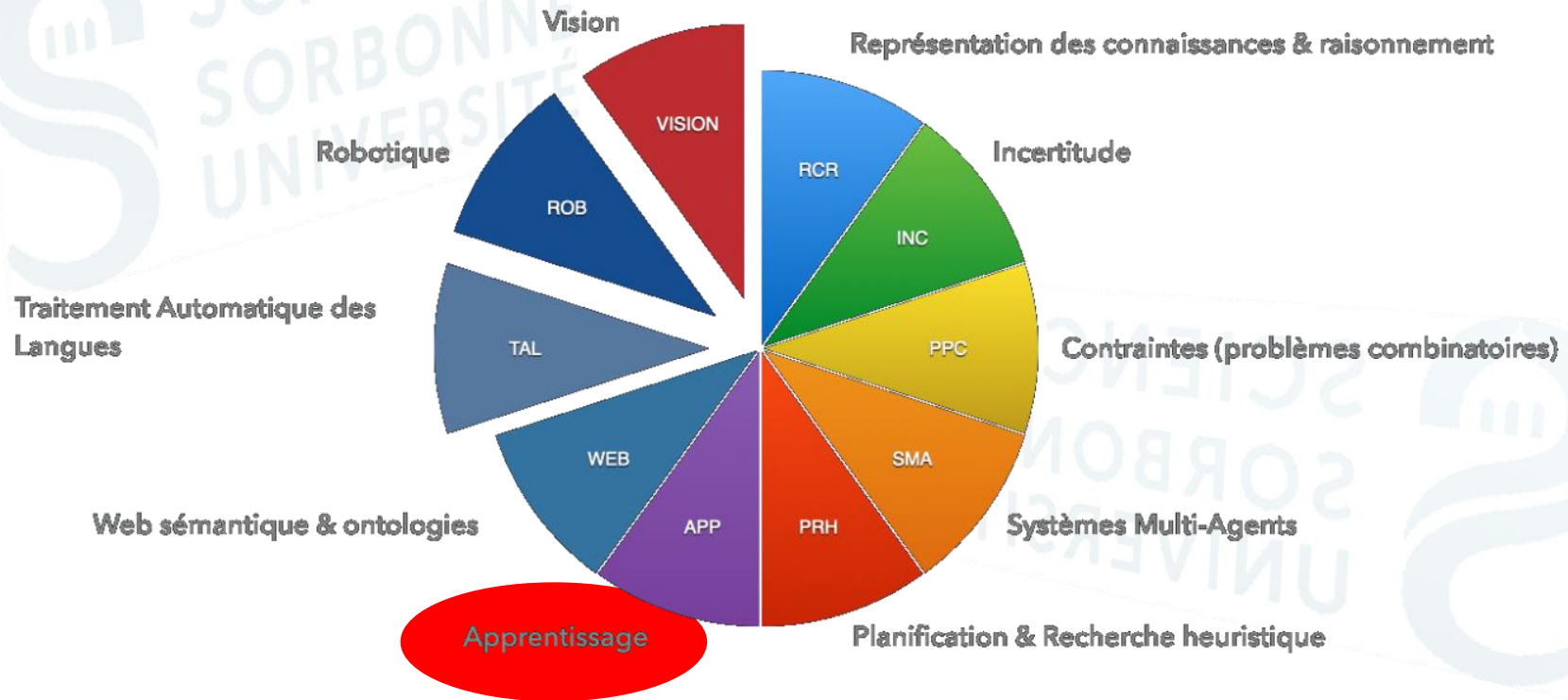
Déroulement

3 séances de cours de 2h

2 séances de TP de (1 de 2h - 1 de 4h)

1 cc sur le cours et les TP

Intelligence artificielle



Les différentes tâches

Classification

- But : L'algorithme doit dire à quelle catégorie appartient une entrée
- Exemple : L'image d'une personne → son identité

Régression

- But : L'algorithme doit associer une valeur numérique à une entrée
- Exemple : Paramètres météorologique → température à 24 heures

Retranscription

- But : A partir de l'observation d'une entrée, la transcrire sous une forme textuelle
- Exemple : A partir d'images de google street, donner les numéros de rue

Traduction

- But : convertir une séquence de symbole en une autre séquence de symbole dans un autre langage
- Exemple : traduction du français à l'anglais

Détection d'anomalie

- But : Trouver si une entrée est atypique ou non
- Exemple : détecter des voitures en sens contraire sur des images routières

Synthèse

- But : apprendre à générer de nouveaux exemples proches de ceux d'une base de données
- Application : Synthétiser des visages sous un autre point de vue

Débruitage

- But : Apprendre à générer des données non bruitées à partir de données bruitées

Applications

Média

- Facebook ou instagram personnalise la diffusion d'informations
- LinkedIn propose des emplois pertinents, des nouveaux contacts personnalisés
- Détection de spam

Objets connectés/intelligents

- Reconnaissance de visages pour appareil photo
- Enceinte intelligente
- Véhicule intelligent : détection de véhicules, de piétons, prédiction de trafic
- Analyse de comportements anormaux
- Assistant personnel

Applications

Finance

- Détection de fraude à la CB
- Prédiction des valeurs financières

Commerce

- Recommandation personnalisée de produit sur internet
- Création de chatbots et de robots pour répondre aux appels téléphoniques des clients

Applications

Transport

- Comment Uber fixe le prix d'une course?
- Comment Uber gère le co-voiturage
- Comment Uber gère le temps d'attente et affecte les véhicules aux clients

Voyage

- 90% des américains partagent leur expérience de voyage
- Tripadvisor reçoit 280 avis par minute
- Tripadvisor analyse les informations pour donner les plus pertinentes

Robotique traditionnelle

- Peu adaptative
- Simple automate qui reproduit des tâches



<https://www.robot-magazine.fr/le-marche-en-pleine-croissance-des-robots-industriels-et-de-lautomatisation>

Robotique collaborative ou cobot

- Réduit les risques de TMS et éliminer la manipulation de charges lourdes
- Améliore l'ergonomie d'un poste de travail
- Rend du personnel disponible pour des opérations à plus forte valeur ajoutée
- Favorise l'expertise et le savoir-faire humain
- **Adaptative et interactive**
- **Réagit à l'environnement**



<https://www.humarobotics.com/industries-faites-le-choix-de-la-robotique-collaborative/>

Véhicule autonome

- Perception de l'environnement
- Collaboration avec l'environnement (feu)
- Collaboration avec les autres usagers



<https://www.pourlascience.fr/sd/transports/itinerare-bis-pour-la-voiture-autonome-23686.php>

Et en robotique, pourquoi de l'intelligence ?

- **Pour percevoir l'environnement**
 - détecter des objets ou des humains
 - se localiser, estimer la pose
 - estimer la profondeur
 - suivre des mouvements
 - Segmenter la scène (robotique chirurgicale)
- **Pour prédire le futur et donc anticiper**
 - Quelle trajectoire va prendre la voiture devant moi, quel geste va réaliser un humain en collaboration avec un robot
- **Pour planifier des actions**
 - Comment réagir suite à une situation
- **Pour interpréter des comportements humains, de voiture, ...**
 - Reconnaissance d'actions pour la collaboration avec un robot, de situation normale ou anormale, d'émotion, du langage
- **Pour générer des mouvements (avatar, véhicule, bras articulé)**
 - Avoir un comportement interactif plutôt que de suivre un ensemble de règle, modéliser les interactions sociales

Difficultés

Considérons le cas de la reconnaissance de visages (classification)

Problème de
résolution

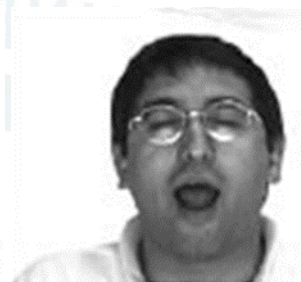


Peut-on définir une distance entre deux visages ?

Problème de
pose



Problème
d'expression
faciale



Problème
d'occultations



MARTINEZ, Aleix M. The AR face database. *CVC Technical Report*24, 1998.

Introduction

Raisonnons sur un cas simple

Reconnaître les truites et les saumons



Codage

Chaque poisson est représenté par un **vecteur de caractéristiques** aussi appelé **codage** ou **features**

Exemple

Supposons que l'on ait N poissons x_i , $1 \leq i \leq N$ de classe y_i , $1 \leq i \leq N$

y_i est la classe $x_i \in \{truite, saumon\}$ de l'exemple i (du poisson i)

x_i est le **vecteur de caractéristiques de dimension n** :

- Si chaque poisson est représenté juste par sa taille, alors $n = 1$ et la base est constituée de N exemples $\{x_i, y_i\}$ où x_i est de dimension 1
- Si chaque poisson est représenté par sa taille et sa teinte, alors $n = 2$ et la base est constituée de N exemples $\{x_i, y_i\}$ où x_i est de dimension 2

Raisonnons sur un cas simple

Classification en 1 dimension : la taille des poissons
→ Les x_i sont de dimension 1

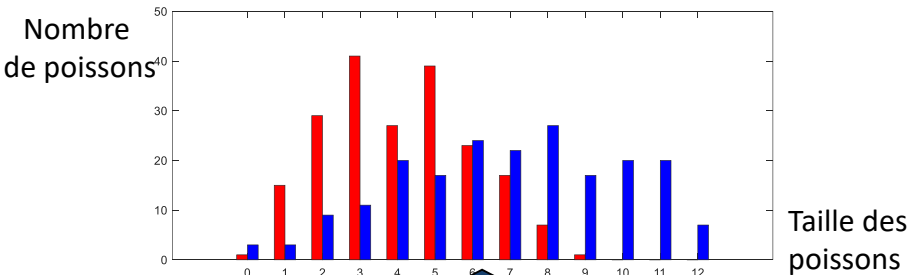
$p(x/\text{truite})$
 \propto histogramme de la taille des truites



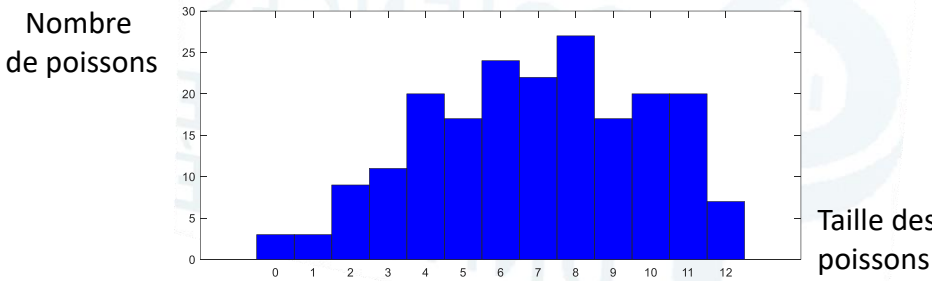
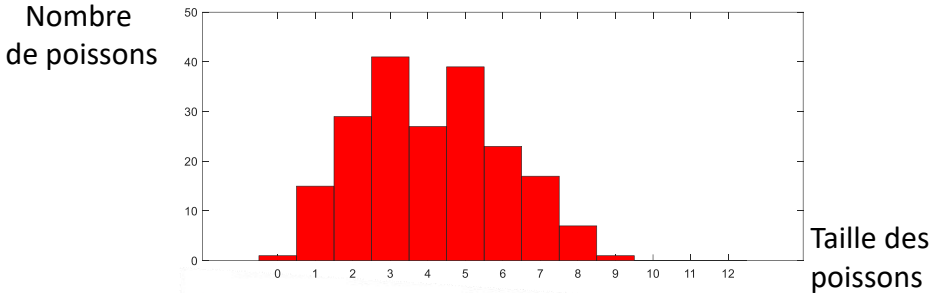
$p(x/\text{saumon})$
 \propto histogramme de la taille des saumons



Les deux superposés



Seuil de décision



Aura-t-on une bonne classification ?

Introduction

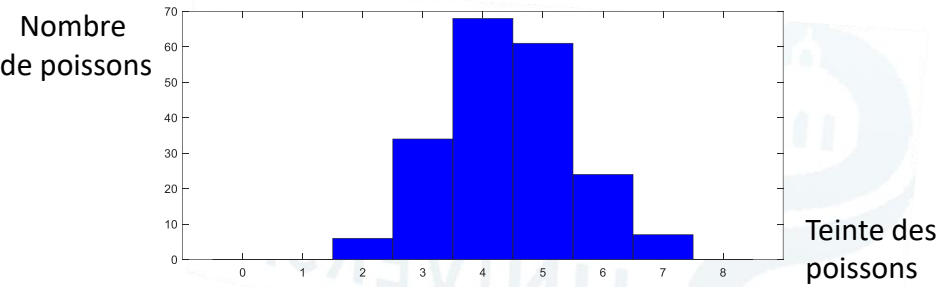
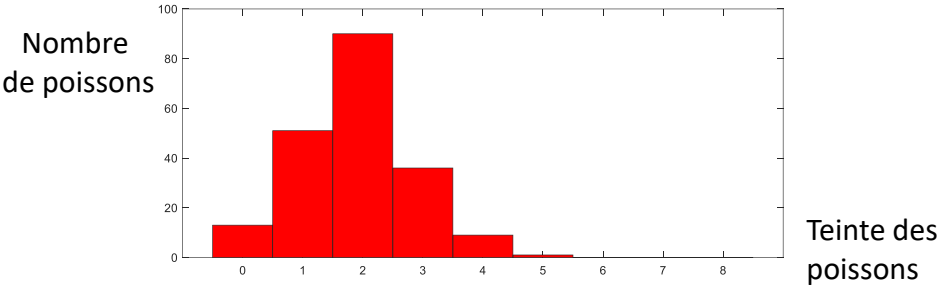
Raisonnons sur un cas simple

Classification en 1 dimension : la teinte des poissons
→ Les x_i sont de dimension 1

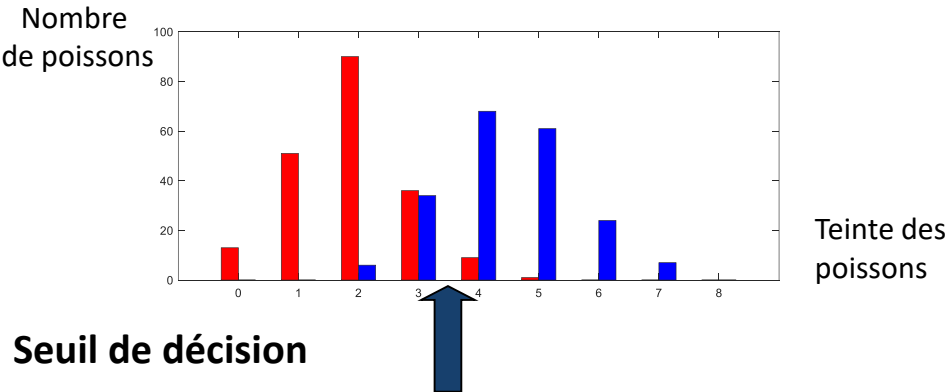
$p(x/\text{truite})$
 \propto histogramme de la teinte des truites



$p(x/\text{saumon})$
 \propto histogramme de la teinte des saumons



Les deux superposés



Est-ce mieux ?

Introduction

Raisonnons sur un cas simple

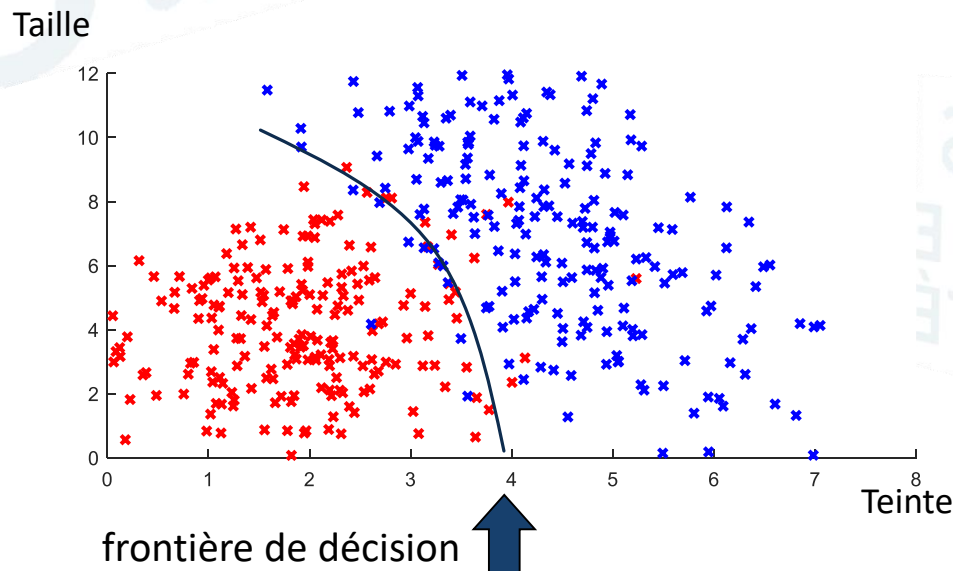
Classification en 2 dimensions : la teinte et la taille des poissons

→ Les x_i sont de dimension 2

Classification avec la teinte et la longueur des poissons

→ Vecteur de caractéristiques à 2 dimensions

→ Chaque poisson est représenté par un point (dimension 2) dans le plan



Il faut considérer les caractéristiques ensemble (et donc pas séparément)

Qu'est ce qu'un bon codage ?

Pouvoir discriminant

- Le codage doit être différent pour des exemples de classes différentes → **forte variance inter-classes**

Pouvoir unifiant

- Le codage doit être à peu près le même pour tous les exemples d'une même classe → **faible variance intra-classe**

Stabilité/invariance

- Codage le plus insensible possible au bruit
- En fonction des applications, invariance en translation, rotation, changement d'échelle

Faible dimension

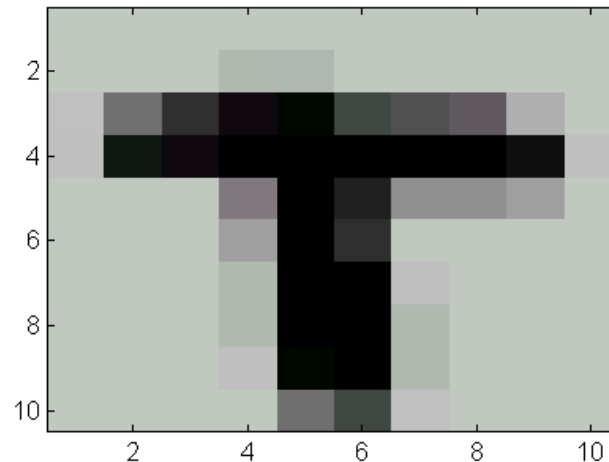
- Plus le codage est de faible dimension, plus les temps de calcul seront faibles
- Augmenter la dimension peut détériorer les résultats de reconnaissance (malédiction des grandes dimensions → compromis à trouver)

Le codage rétinien

Mais on est vite en grande dimension

Exemple pour des lettres représentées par un codage rétinien de taille 10x10

→ Vecteur de caractéristiques de dimension 100



Classifier

= associer une **classe** C

à un **vecteur de caractéristiques** x de dimension n

Base de données divisée en 3

- **Base d'apprentissage** : pour apprendre le modèle
- **Base de validation** : pour aider à définir certains paramètres du modèle
- **Base de test** : pour tester le modèle sur de nouvelles données jamais rencontrées

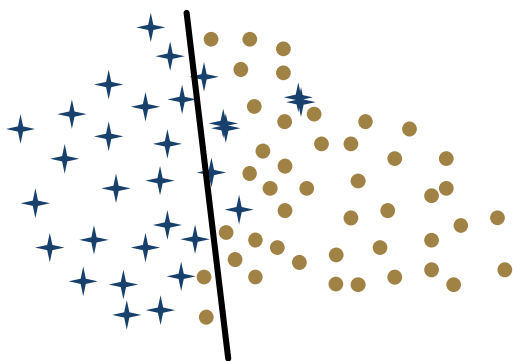
Hypothèse :

La distribution des données d'apprentissage est représentative de celle des données de test

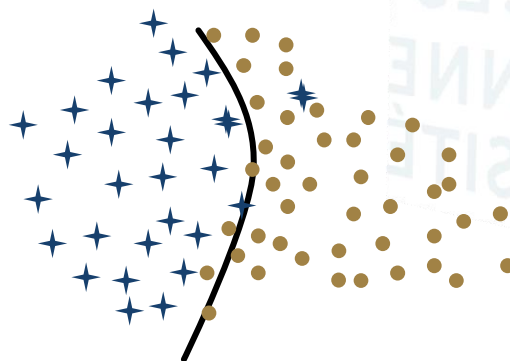
Un bon apprentissage devra :

- Avoir une faible erreur sur la base d'apprentissage
- Avoir un écart faible entre l'erreur d'apprentissage et l'erreur de test

Lorsque c'est le cas, on parle de **bonne généralisation** : le système est capable de reconnaître des données jamais vues



Sous-apprentissage

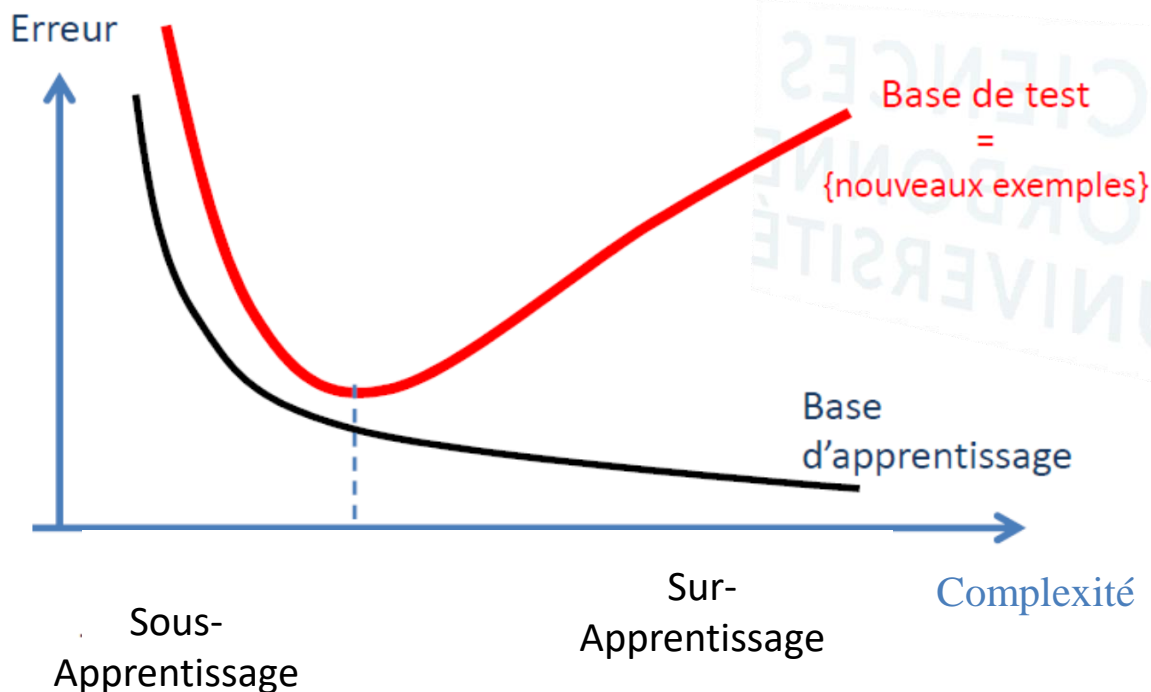


Sur-apprentissage

→ Il faut estimer les paramètres du modèle

Problème :

- Les meilleures performances sur la base d'apprentissage ne correspondent pas au meilleur modèle
- Données : vrai modèle + bruit. Les meilleures performances sur la base d'apprentissage sont obtenues quand on a appris le bruit



Sur-apprentissage (over-fitting)

On se fie trop aux données

- petit changement de la base d'apprentissage = forte variation de la sortie
- **Variance élevée, biais faible**

Sous-apprentissage (under-fitting)

On se fie moins aux données

- on s'éloigne par endroit de la vraie frontière
- **Biais élevé, variance faible**

Compromis biais/variance

Un peu de théorie

Vecteur de caractéristiques x

Variable à prédire $y = f(x) + \varepsilon$ où ε bruit de moyenne nulle et de variance σ^2

➔ Base de données : N exemples (x_i, y_i) tirés selon \mathcal{P} : $\{x_i, y_i\}_{1 \leq i \leq N}$

Qualité d'un modèle :

On a un modèle qui prédit $\hat{y}_i = \hat{f}(x_i)$

On dispose d'une fonction perte (erreur) : $l(\hat{f}(x), y)$

Risque $L(\hat{f}) = E_{(x,y) \sim \mathcal{P}} [l(\hat{f}(x), y)]$

- Modèle avec les meilleures performances $f^* = \underset{\hat{f}}{\arg \min} (E_{(x_i, y_i)} [l(\hat{f}(x_i), y_i)])$
- Modèle recherché $f = \underset{\hat{f}}{\arg \min} L(\hat{f}) = E_{(x,y) \sim \mathcal{P}} [l(\hat{f}(x), y)]$

Pb : on apprend le bruit

Cas de la régression $l(\hat{f}(\mathbf{x}), y) = (y - \hat{f}(\mathbf{x}))^2$

$$\text{Risque } L(f) = E_{(x,y) \sim \mathcal{P}} [l(\hat{f}(\mathbf{x}), y)] = E \left[(f(\mathbf{x}) + \varepsilon - \hat{f}(\mathbf{x}))^2 \right]$$

$$= E \left[(f(\mathbf{x}) + \varepsilon - \hat{f}(\mathbf{x}) + E[\hat{f}(\mathbf{x})] - E[\hat{f}(\mathbf{x})])^2 \right]$$

$$= E \left[(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})] + E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}) + \varepsilon)^2 \right]$$

$$= E \left[(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])^2 + (E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}))^2 + (\varepsilon)^2 + 2\varepsilon(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})]) - 2\varepsilon(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x})) - 2(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x})) \right]$$

$$= E \left[(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])^2 \right] + E \left[(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}))^2 \right] + E[(\varepsilon)^2] + \underbrace{2E[\varepsilon(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])]}_0 - \underbrace{2E[\varepsilon(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}))]}_0 - \underbrace{2E[(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}))]}_0$$

$$= E \left[(f(\mathbf{x}) - E[\hat{f}(\mathbf{x})])^2 \right] + E \left[(E[\hat{f}(\mathbf{x})] - \hat{f}(\mathbf{x}))^2 \right] + E[(\varepsilon)^2]$$

Risque = biais² + variance + bruit

$$\text{Erreur} = \text{biais}^2 + \text{variance} + \text{bruit}$$

► **Biais** : $E \left[\left(f(x) - E[\hat{f}(x)] \right)^2 \right]$

- erreur provenant d'hypothèses erronées
- manque de cohérence entre les données en entrée et les sorties (sous-apprentissage).
- différence entre la moyenne des valeurs prédites (valeurs obtenues à partir de différents ensembles d'apprentissage) et la valeur cible au point x .

► **Variance** : $E \left[\left(E[\hat{f}(x)] - \hat{f}(x) \right)^2 \right]$

- Sensibilité aux fluctuations des échantillons de la base d'apprentissage (sur-apprentissage).
- Espérance de la différence entre la valeur prédite pour x et la moyenne de toutes les valeurs prédites (obtenues à partir de différents ensembles d'apprentissage).

► **Bruit** : $E[(\varepsilon)^2]$

- Aussi appelé erreur irréductible car présent dans les données $y = f(x) + \varepsilon$

En pratique,

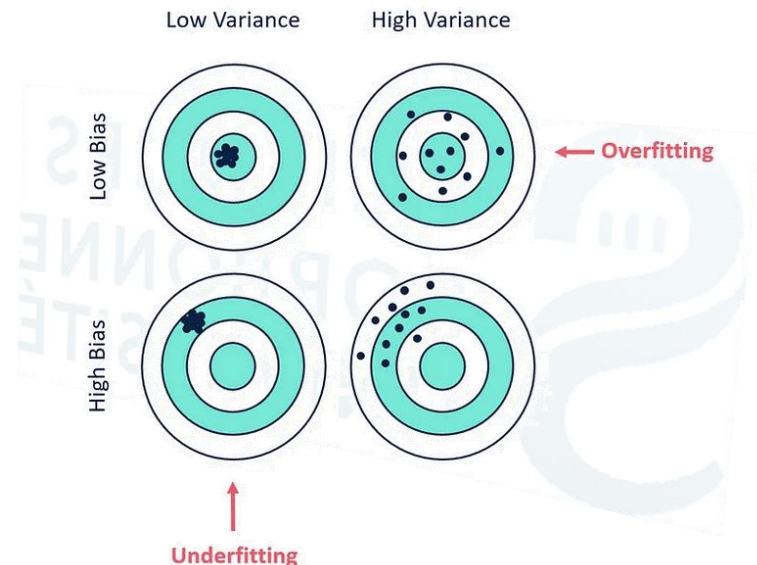
- On décompose la base d'apprentissage en Nb sous-ensembles.
- Sur chacun des sous-ensembles j , on estime $\hat{f}_j()$
- Pour chaque exemple (\mathbf{x}, y) , Nb prédictions $\hat{f}_j(\mathbf{x})$

Biais : $\overline{\hat{f}(\mathbf{x})} - y$: écart entre la moyenne des prédictions et la vérité de terrain

Variance : variance des prédictions $\hat{f}_j(\mathbf{x})$

- On moyenne sur tous les exemples

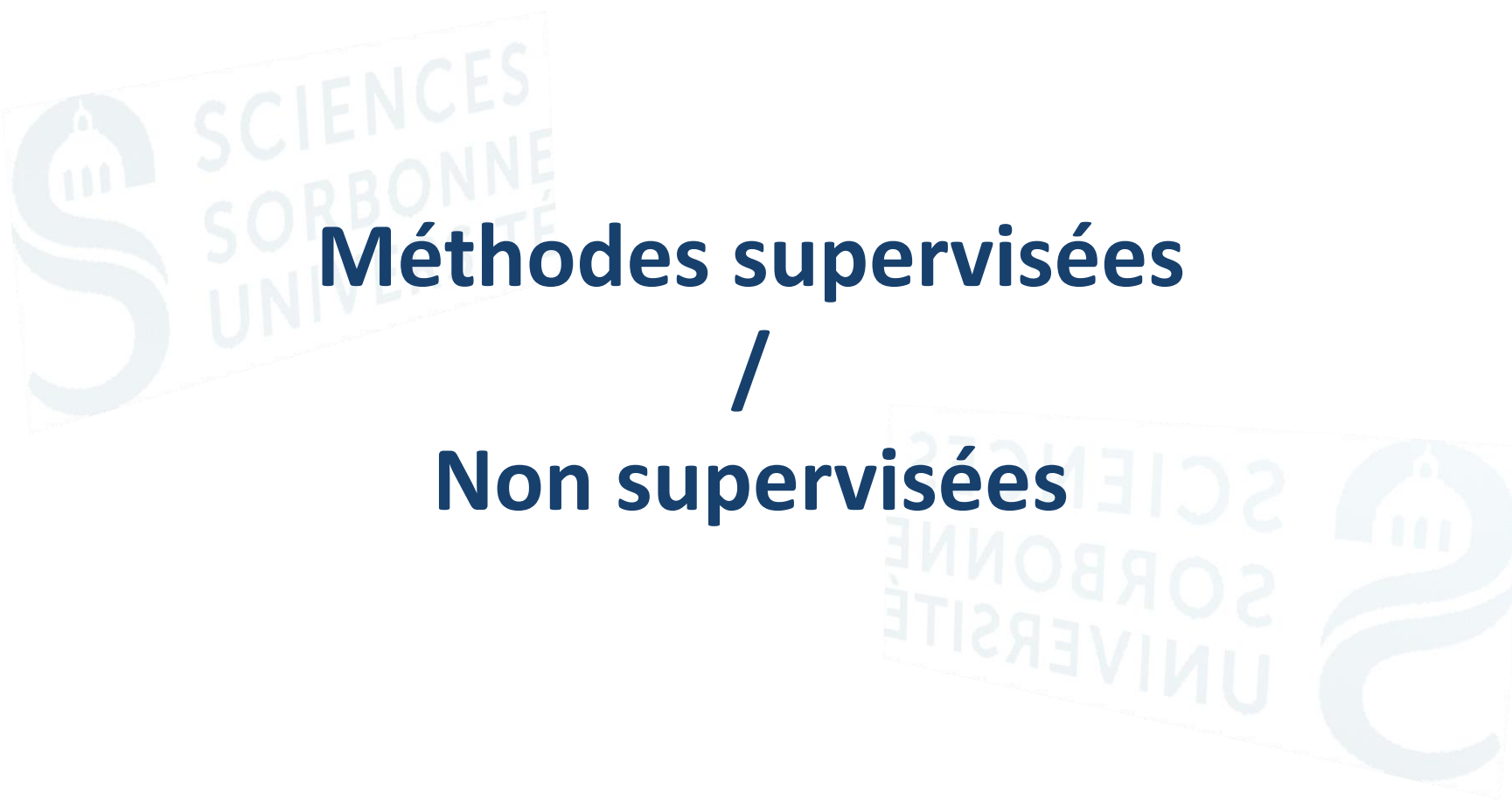
- Si le **biais est trop grand**, besoin d'un modèle plus complexe ou de plus d'itérations
- Si la **variance est trop grande**, besoin d'un modèle moins complexe ou de moins d'itérations



Comment avoir un bon compromis biais/variance ?

Comment avoir une bonne généralisation

- En diminuant/augmentant la dimension des exemples
- En augmentant le nombre d'exemples
- En sélectionnant un bon modèle (complexité) ou en le contraignant (régularisation)
- Avec des méthodes ensemblistes (il vaut souvent mieux recueillir plusieurs avis que de se fier à un seul → on apprend sur plusieurs ensembles de données puis vote majoritaire par exemple)



Méthodes supervisées / Non supervisées

Méthode supervisée

On dispose d'un **ensemble d'exemples étiquetés** en apprentissage

➔ On souhaite savoir classer un nouvel exemple

Le nombre de classes est connu *a priori*

Exemple

On dispose d'un ensemble de signaux audio enregistré à partir de 50 personnes. L'identité de la personne est connue pour chaque enregistrement

Pour un nouveau signal, on souhaite déterminer l'identité de la personne

Méthode non supervisée

On dispose d'un **ensemble d'exemples non étiquetés** en apprentissage

➔ On souhaite partitionner cet ensemble en sous-ensembles homogènes

Le nombre de classes (sous-ensembles) n'est pas connu *a priori*

Exemple

On dispose des achats de livres de clients sur amazon

On souhaite catégoriser les clients en fonction de leur goûts afin de leur proposer des nouveaux achats pertinents

➔ On peut utiliser ces méthodes pour explorer et comprendre les données



Méthodes génératives / discriminatives

Soit \mathbf{x} les exemples de dimension n et y leur classe telle que $y \in [1, \dots, K]$

But de la classification : déterminer $p(y|\mathbf{x})$

Approche discriminative

Détermine directement $p(y|\mathbf{x})$

Approche générative / Classification bayésienne :

Détermine pour chaque classe $p(\mathbf{x}|y)$ et $P(y)$ puis utilise le théorème de Bayes :

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

Où le dénominateur est un terme de normalisation :

$$p(\mathbf{x}) = \sum_y p(\mathbf{x}|y)p(y)$$

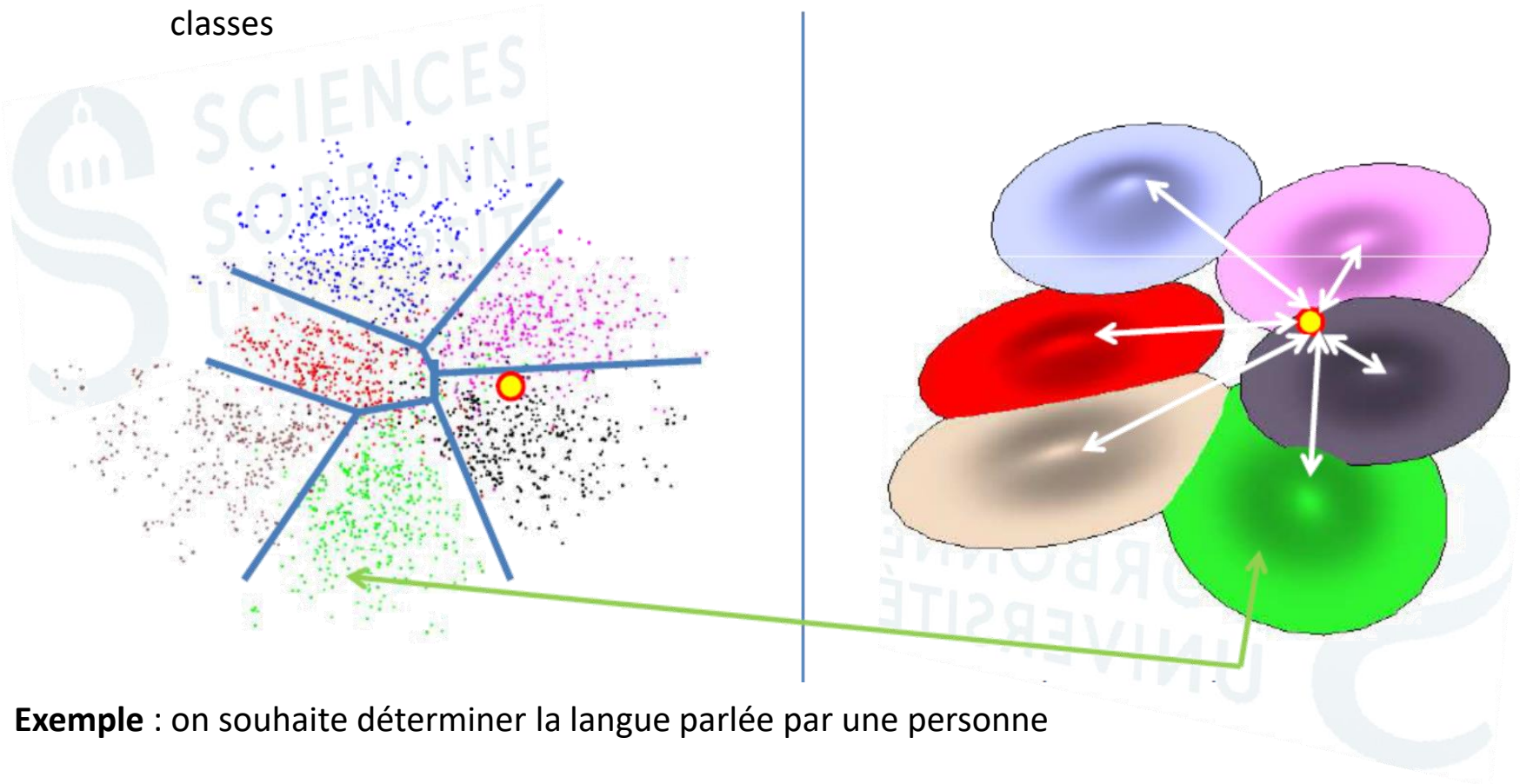
Cette approche est dite générative car, connaissant $p(\mathbf{x}|y)$, il est facile de générer des données dans l'espace des paramètres

Approche discriminante

On apprend les frontières entre les classes

Approche générative

On modélise chaque classe



Exemple : on souhaite déterminer la langue parlée par une personne

Approche générative : on apprend chaque langue puis on détermine à quel langue la parole appartient (peut fonctionner avec une seule langue pour savoir si la personne parle français ou non)

Approche discriminative: on apprend les différences linguistiques entre les langues, sans apprendre la langue. Beaucoup plus simple !

Avantage/inconvénient

Approche générative



- $p(x|y)$ est estimée. On peut considérer $p(x|y)$ comme la probabilité que x soit bien modélisé par le modèle. Ceci permet de faire du rejet ou de combiner des classifieurs.
- $p(x|y)$ peut être utilisé pour générer des données
- Permet à un système d'utiliser une seule classe. Ex : la teinte chair



- Trouver $p(x|y)$ pour chaque classe est très coûteux en temps de calcul, surtout quand x est de grande dimension
- Nécessite une grande base de données, surtout quand x est de grande dimension

Approche discriminative



- Il est beaucoup plus rapide de déterminer $p(y|x)$ car la dimension de y est bien plus faible que celle de x



- On ne peut pas générer de données
- On ne peut pas faire de rejet
- Difficile de combiner des classifieurs

Méthodes génératives	Méthodes discriminatives
Classification bayésienne	K plus proches voisins
HMM (Hidden Markov Model)	Arbres de décision
Réseaux bayésiens	SVM (Support Vector Machine)
MRF (Markov Random Fields)	RVM (Relevance Vector Machine)
	Réseaux de neurones
	CRF (Conditional Random fields)

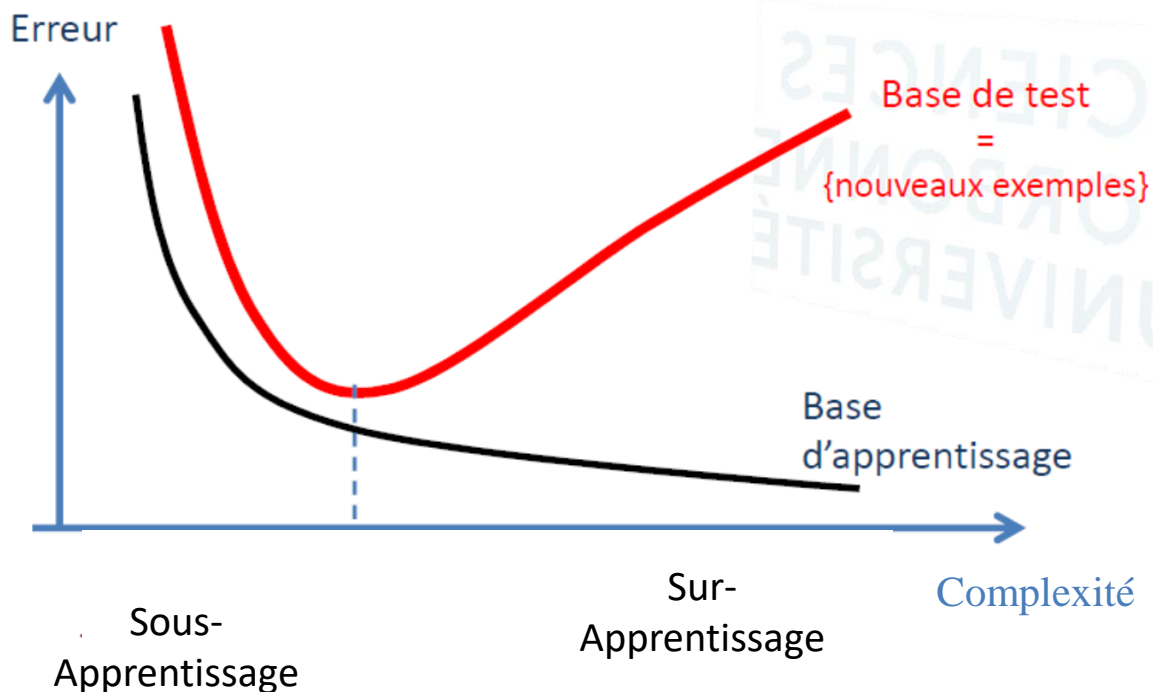


Performances d'un classifieur

→ Il faut estimer les paramètres du modèle

Problème :

- Les meilleures performances sur la base d'apprentissage ne correspondent pas au meilleur modèle
- Données : vrai modèle + bruit. Les meilleures performances sur la base d'apprentissage sont obtenues quand on a appris le bruit

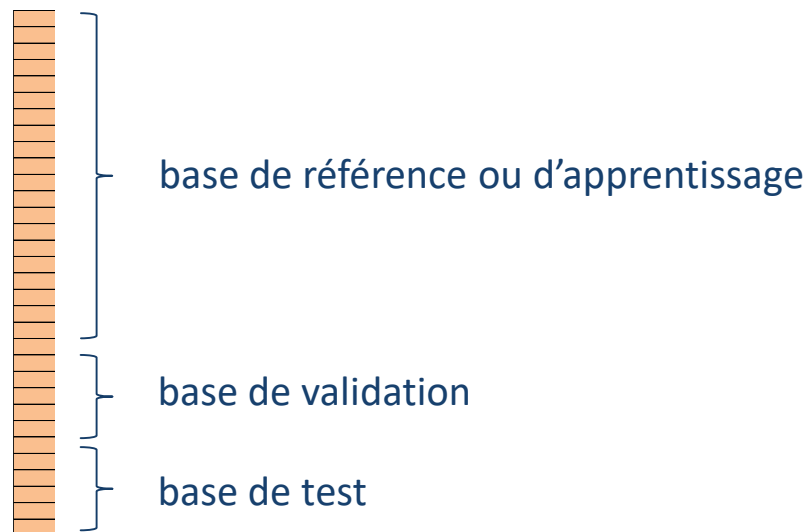


En apprentissage, il n'y a **pas de méthode idéale**, celle-ci varie en fonction des données, de leur dimension, du problème posé,...

- Le choix de la méthode la plus adaptée à un cas précis repose sur une **estimation de l'erreur**
- Cette estimation devra être la plus rigoureuse possible

On utilise 3 bases :

- **Une base de référence ou d'apprentissage** utilisée pour mettre au point le classificateur
- **Une base de validation** pour déterminer les paramètres du classifieur
- **Une base de test** : exemples jamais vus au préalable pour évaluer le classificateur



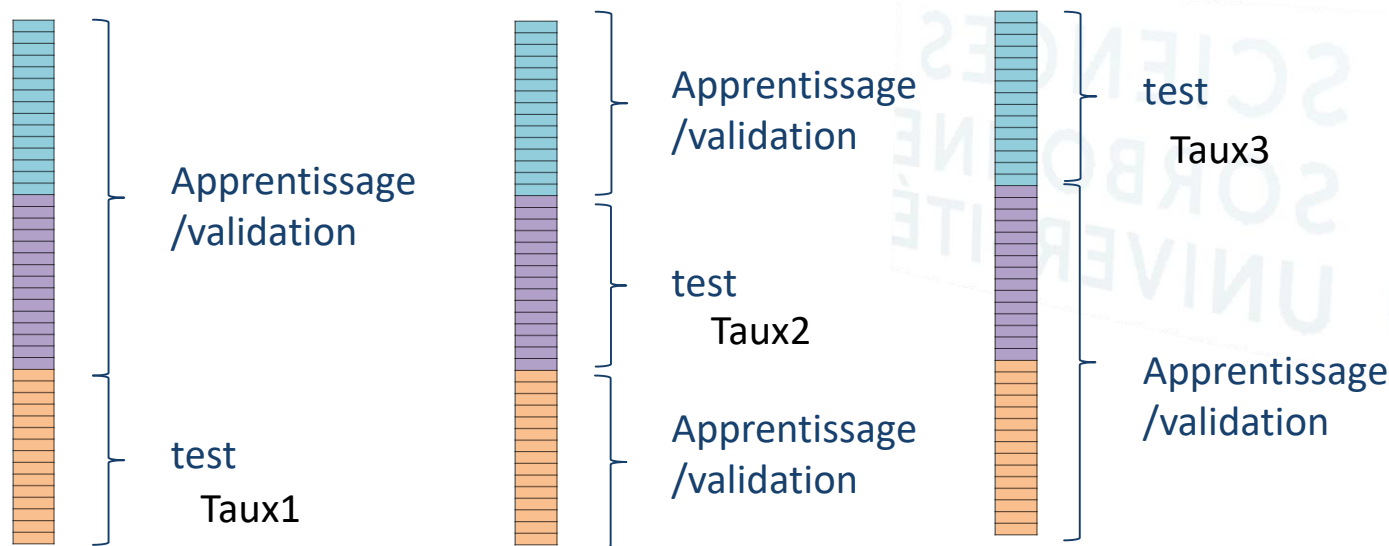
Petite base de données

Avec peu de données, il est difficile d'estimer les performances de manière fiable.

Solution 1 : K-fold validation :

- Division de la base en K sous-ensembles.
- (K-1) sous-ensembles pour l'apprentissage et la validation, le dernier en test
- On fait tourner le sous-ensemble de test
- Le taux de reconnaissance final est la moyenne des taux

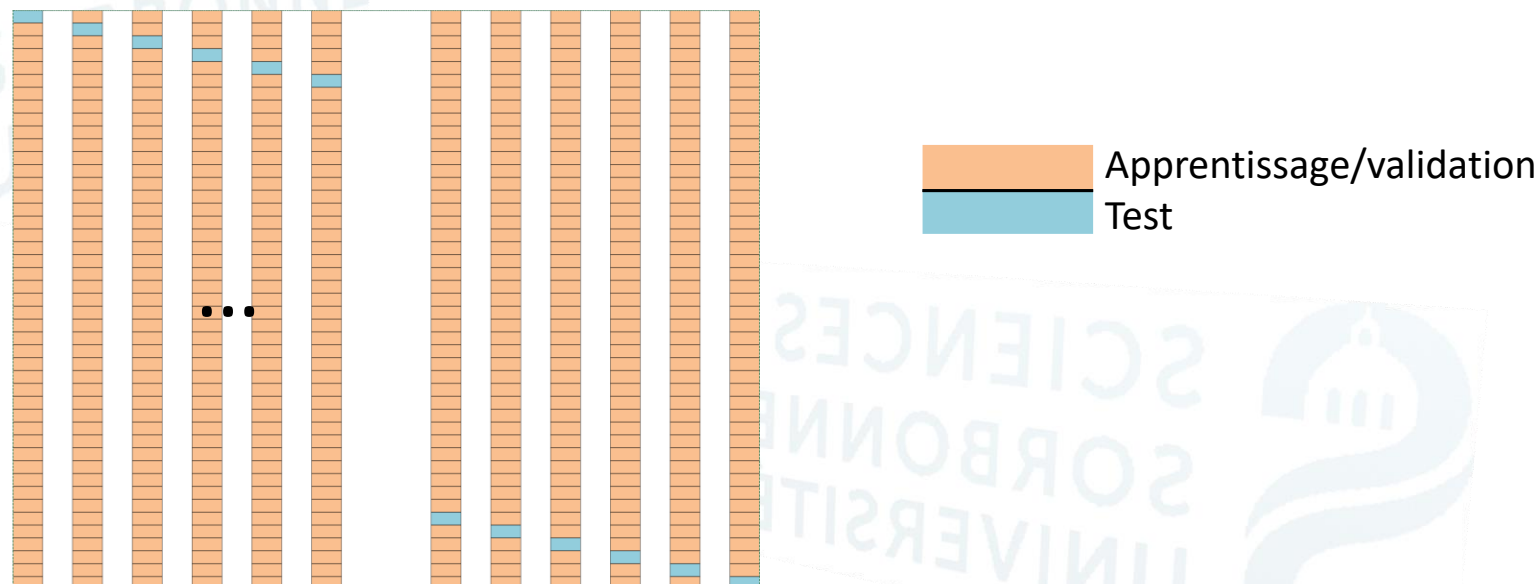
Exemple avec K=3



$$\text{Taux} = (\text{Taux1} + \text{Taux2} + \text{Taux3})/3$$

Solution 2 : Leave-One-Out Cross-Validation (LOOCV)

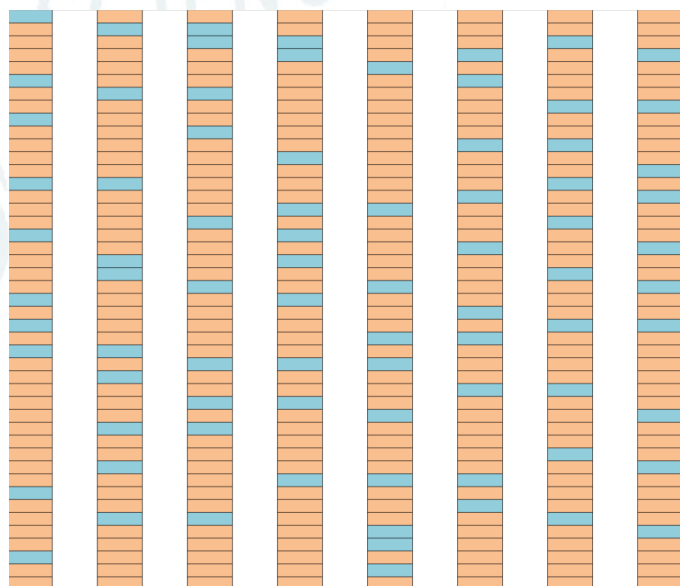
- C'est une K-fold validation dans le cas particulier où $K = \text{nombre d'exemples}$
- $N-1$ exemples pour l'apprentissage et la validation, le dernier exemple en test
- On fait tourner l'exemple de test sur tous les exemples de la base



- Il faut faire autant d'apprentissages qu'il y a d'exemples de test → très long
- Taux de reco = $\text{Nb d'exemples bien classés} / \text{Nb d'exemples}$

Solution 3 : bootstrap

- Tirage aléatoire pour déterminer les exemples d'apprentissage et de test
- Plusieurs itérations pour déterminer plusieurs sous-ensembles



Apprentissage
Test

- A chaque fois on calcule le taux de reconnaissance sur les exemples de test
- Le taux de reconnaissance final est le taux moyen

ATTENTION :

- Quand les données sont très déséquilibrées, tenir compte des classes lors du partitionnement
 - Chaque sous-ensemble d'apprentissage/validation et test doit contenir des exemples de chaque classe
- Quand les données sont partitionnées, isoler les partitions de test pour éviter les corrélations
 - Par exemple, 100 exemples pour une même expérience
 - Chaque personne réalise l'expérience 10 fois
 - Mettre les 10 exemples de la même personne en test pour éviter les corrélations (l'identité de la personne de test ne doit pas être présente en apprentissage)

Problèmes binaires – très utilisés pour les problèmes de détection

2 classes (**positif et négatif**)

Les positifs correspondent aux objets à détecter

On définit les :

- Vrai Positif (True Positive)
- Vrai Négatif (True Négatif)
- Faux Négatif (False Négatif)
- Faux Positif (False Positif)

Classe trouvée →

	Classe réelle ↓	
	+	-
+	TP	FN
-	FP	TN

Problèmes binaires – très utilisés pour les problèmes de détection

2 classes (**positif et négatif**)

Les positifs correspondent aux objets à détecter

$$\text{Accuracy} = \text{Taux de bonne reconnaissance} = \frac{TP+TN}{TP+TN+FP+FN} \times 100$$

= Taux de reconnaissance

Mesure pas suffisante, ne précise pas comment on se trompe.

Fausse détection ?

Manque de détection ?

Classe trouvée →

	Classe réelle ↓	
	+	-
+	TP	FN
-	FP	TN

Taux de bonne classification - accuracy

Problème :

Le taux de bonne classification est une mesure faible qui **ne tient pas compte de la distribution des classes**

→ Quand les classes sont très disproportionnées, prédire systématiquement la classe majoritaire amène à un bon taux de classification

Exercice :

En diagnostic médical, très peu de personnes sont malades (5%?). Quel est le taux de classification si on prédit toujours que la personne est saine ?

Exemple sur 100 personnes

	malade	sain
malade	0	5
sain	0	95

Tb=95%

Solution :

On tient compte de la répartition des classes et on construit une **matrice de confusion normalisée**

	e_{11}/N_1	e_{12}/N_1
	e_{21}/N_2	e_{22}/N_2

N_1 : Nombre d'exemples de la classe 1

N_2 : Nombre d'exemples de la classe 2

Le nouveau taux de bonne classification devient

$$tb = \frac{1}{N_c} \sum_{i=1}^{N_c} \frac{e_{ii}}{N_i} \text{ où } N_c \text{ est le nombre de classes}$$

Exercice :

reprendre l'exemple précédant avec cette normalisation

On a maintenant

	malade	sain
malade	0	1
sain	0	1

Et $t_b = 50\%$

Précision/Rappel

Précision = $\frac{TP}{TP+FP} = \frac{\text{Nb positifs bien classés}}{\text{Nb classés positifs}}$
= capacité du système à ne pas détecter de faux positifs

Rappel = $\frac{TP}{TP+FN} = \frac{\text{Nb positifs bien classés}}{\text{Nb positifs}}$
= capacité du système à détecter les positifs

	+	-
+	TP	FN
-	FP	TN

Un bon classifieur doit avoir une précision et un rappel de 1

Souvent ces deux notions sont contradictoires

Bonne précision \Leftrightarrow mauvais rappel

Bon rappel \Leftrightarrow mauvaise précision

Combiner deux mesures en une:

$$\textbf{F1-score} = \frac{2 \times \text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Précision/Rappel

$$\text{Précision} = \frac{TP}{TP+FP}$$

$$\text{Rappel} = \frac{TP}{TP+FN}$$

$$\text{F1-score} = \frac{2 \times \text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Exercice avec 4 classifieurs

	+	-
+	100	0
-	0	100

	+	-
+	100	0
-	100	0

	+	-
+	80	20
-	20	80

	+	-
+	20	80
-	10	90

Précision				
Rappel				
F1-score				
Accuracy				

Précision/Rappel

$$\text{Précision} = \frac{TP}{TP+FP}$$

$$\text{Rappel} = \frac{TP}{TP+FN}$$

$$\text{F1-score} = \frac{2 \times \text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Exercice avec 4 classifieurs

	+	-
+	100	0
-	0	100

	+	-
+	100	0
-	100	0

	+	-
+	80	20
-	20	80

	+	-
+	20	80
-	10	90

Précision	1 Les détectés + sont bien des +	0.5 50% des détectés + sont des +	0.8 80% des détectés + sont des +	0.66 66% des détectés + sont des +
Rappel	1 On a bien détecté tous les +	1 On a bien détecté tous les +	0.8 On a détecté 80 % des +	0.2 On a détecté 20 % des +
F1-score	1	0.66	0.8	0.3
Accuracy	1	0.5	0.8	0.55

Courbe ROC (Receiver Operating Characteristic)

$$\text{Sensibilité} = \frac{TP}{TP+FN} = \frac{\text{Nb positifs bien classés}}{\text{Nb positifs}}$$

$$\text{Spécificité} = \frac{TN}{FP+TN} = \frac{\text{Nb négatifs bien classés}}{\text{Nb négatifs}}$$

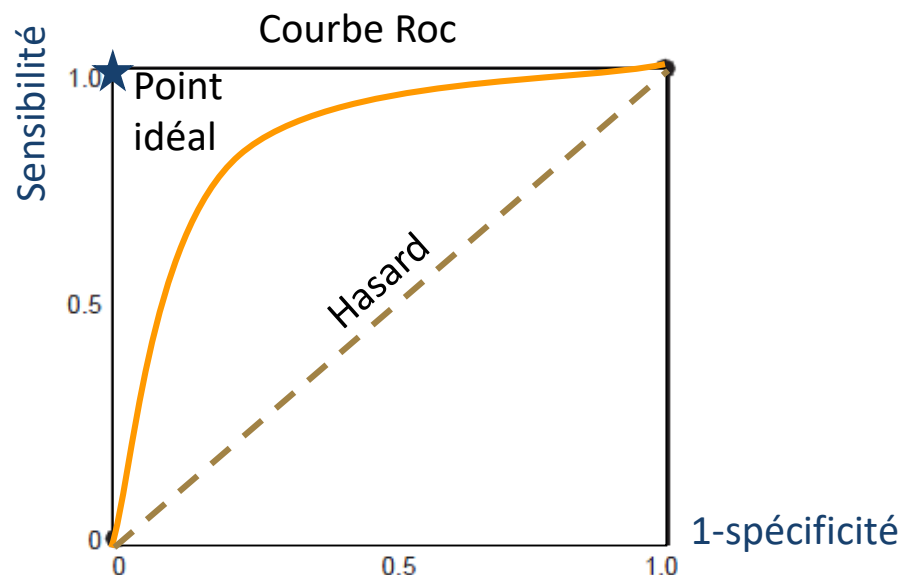
	+	-
+	TP	FN
-	FP	TN

Un bon classificateur devra être

- sensible : détecter les positifs = pourcentage de vrais positifs détectés
- spécifique : ne pas tout détecter comme positif = pourcentage de vrais négatifs détectés

Généralement, plus un classificateur est sensible, moins il est spécifique et vice versa

En faisant varier le seuil de détection, on obtient plusieurs seuils sur la courbe ROC



Courbe ROC

$$\text{Sensibilité} = f(1-\text{spécificité})$$

Toutes les courbes ROC passent par l'origine et par le point (1,1)

On utilise souvent l'aire sous la courbe ROC pour évaluer la performance

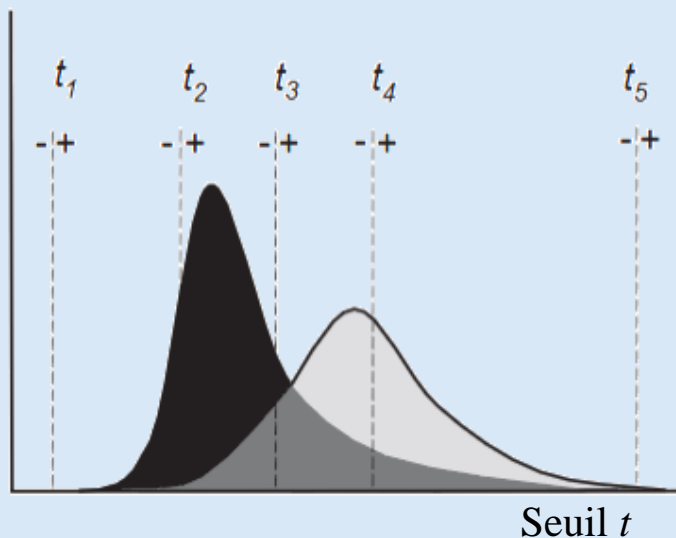
AUC : Area Under Curve

Exercice

On souhaite réaliser une classification binaire. Plusieurs algorithmes dépendant d'un seuil doivent être comparés.

Un algorithme donné donne les densités de probabilité de chaque classe en fonction du seuil.

Tracer la courbe ROC correspondante

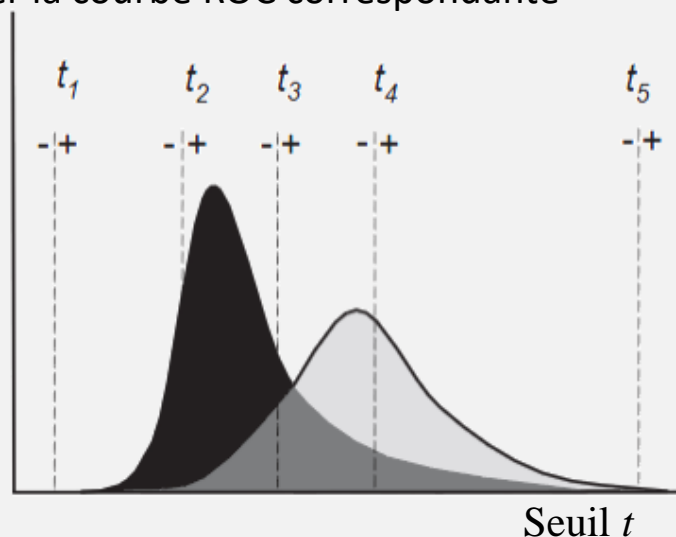


Exercice

On souhaite réaliser une classification binaire. Plusieurs algorithmes dépendant d'un seuil doivent être comparés.

Un algorithme donné donne les densités de probabilité de chaque classe en fonction du seuil.

Tracer la courbe ROC correspondante

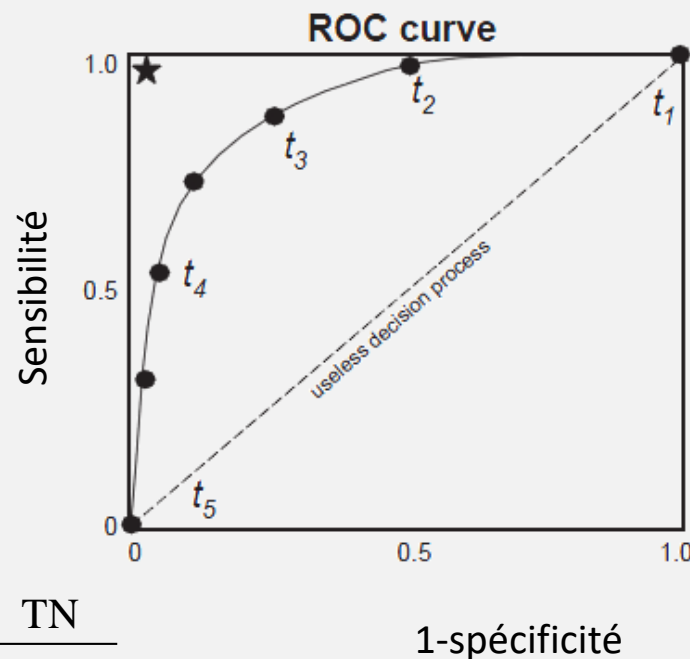


$$\text{Sensibilité} = \frac{TP}{TP + FN}$$

$$\text{Spécificité} = \frac{TN}{TN + FP}$$

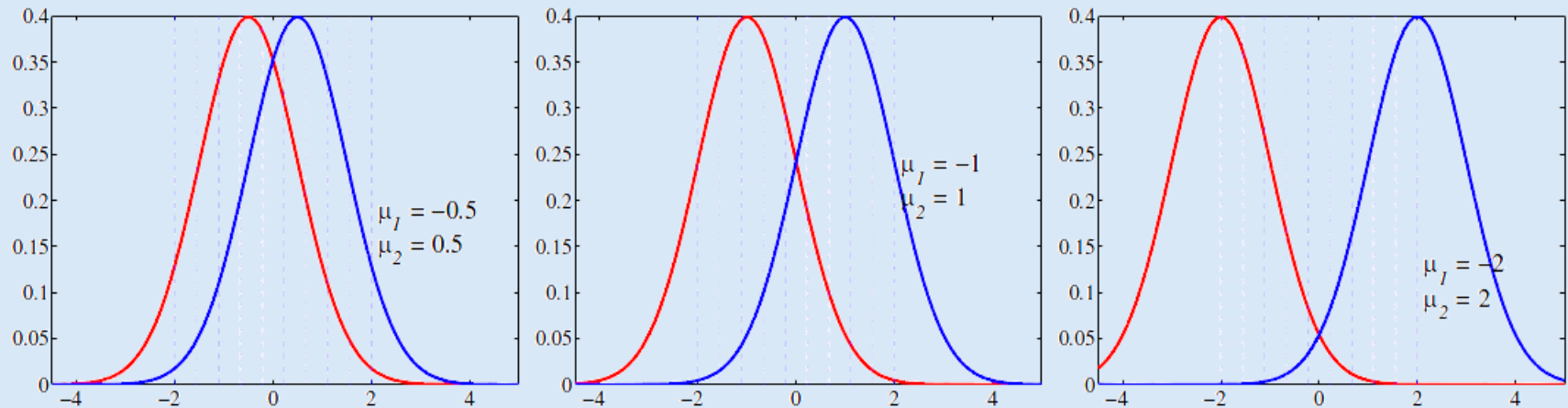
Remarque:

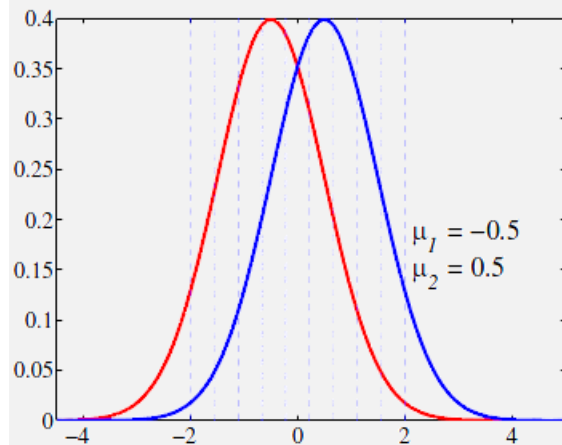
$$\begin{cases} TP + FN = cte \\ TN + TP = cte \end{cases}$$



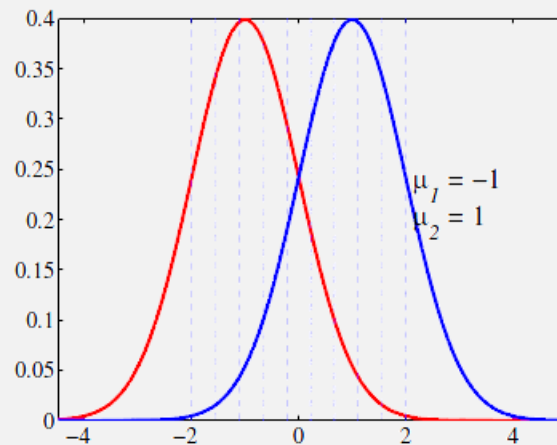
Exercice

Tracer l'allure des courbes ROC des 3 classifieurs ci-dessous (mêmes figures que pour l'exercice précédent))

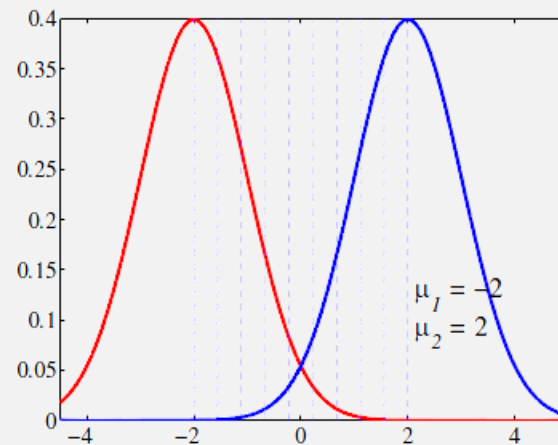




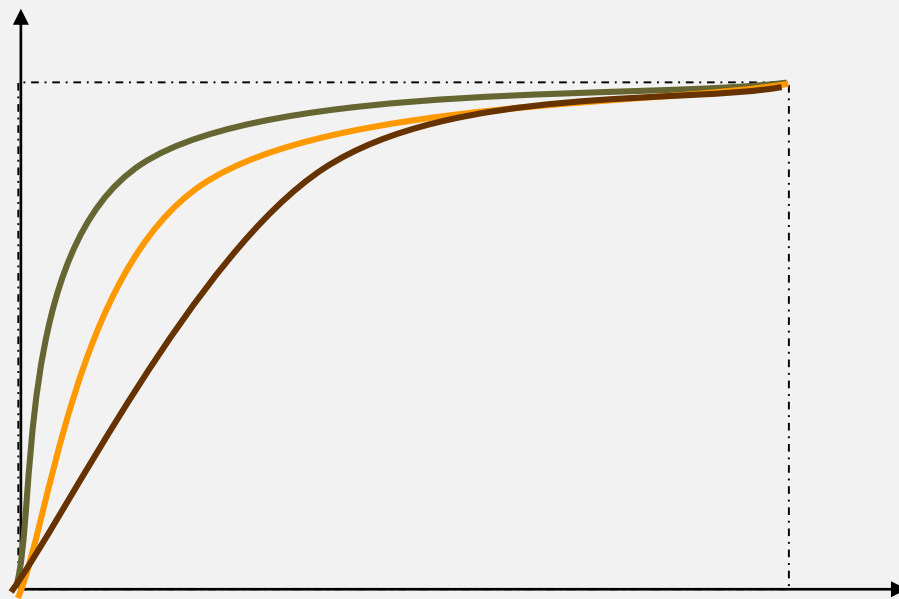
Classifieur 1



Classifieur 2



Classifieur 3



Exercice

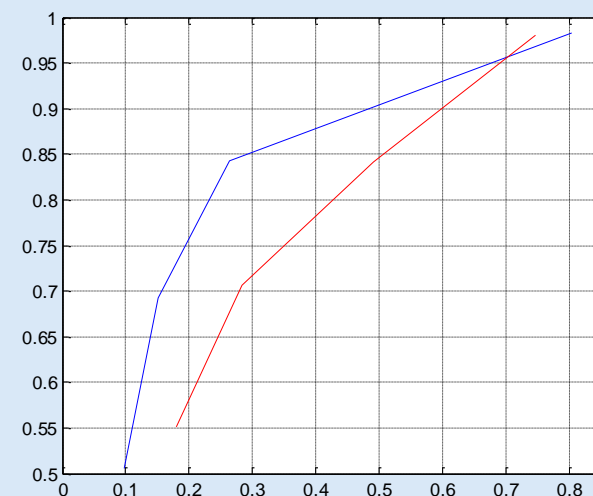
Nous souhaitons détecter des objets en mouvement (positif) dans des séquences d'images.

→ 2 algorithmes que l'on veut comparer indépendamment du seuil de détection

A = Nombre de pixels détectés en mouvement
B = Nombre de pixels détectés non mouvement
C = Nombre de pixels réellement en mouvement et détecté en mouvement
D = Nombre de pixels réellement en mouvement = 57587

		A	B	C
Première méthode	Seuil1	372417	77583	56629
	Seuil2	151899	298101	48526
	Seuil3	98997	351003	39847
	Seuil4	67181	382819	29135
Deuxième méthode	Seuil1	349806	100194	56479
	Seuil2	241185	208815	48472
	Seuil3	151902	298098	40654
	Seuil4	102264	347736	31739

1. Donner l'expression analytique du nombre de vrais positifs (TP), vrais négatifs (TN), faux positifs (FP) et faux négatifs (FN) en fonction de A, B, C, D.
2. Dans la base de test, quel est le nombre d'exemples positifs et d'exemples négatifs ? Donner l'expression analytique et le résultat numérique.
3. Donner l'expression analytique de la sensibilité et de la spécificité en fonction de A, B, C et D.
4. Les deux courbes ROC sont représentées ci-contre. Quelle est la courbe ROC de la première méthode ? Justifier votre réponse. Quelle méthode donne les meilleurs résultats ?



1. Donner l'expression analytique du nombre de vrais positifs (TP), vrais négatifs (TN), faux positifs (FP) et faux négatifs (FN) en fonction de A, B, C, D.

décision \ étiquette	+	-
+	TP	FN
-	FP	TN

$$\begin{aligned}
 A &= TP + FP \\
 B &= TN + FN \\
 C &= TP \\
 D &= TP + FN
 \end{aligned}
 \Rightarrow
 \begin{cases}
 TP = C \\
 FP = A - C \\
 FN = D - C \\
 TN = B - D + C
 \end{cases}$$

2. Dans la base de test, quel est le nombre d'exemples positifs et d'exemples négatifs ?
Donner l'expression analytique et le résultat numérique.

$$\text{Nb ex} > 0 = TP + FN = C + D - C = D = 57587$$

$$\text{Nb Ex} < 0 = TN + FP = A - C + B - D + C = A + B - D = 392413$$

3. Donner l'expression analytique de la sensibilité et de la spécificité en fonction de A, B, C et D.

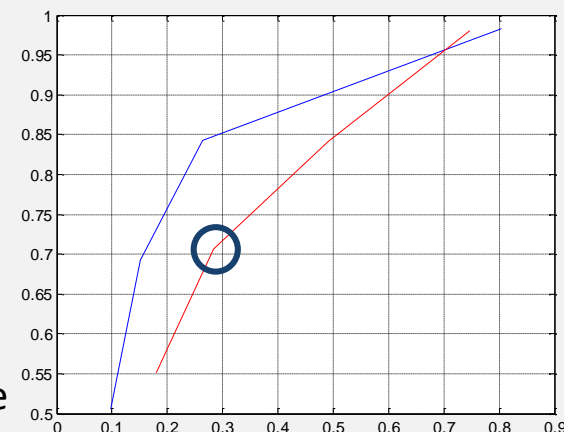
$$\text{sens} = \frac{C}{D} = \frac{C}{57587} \quad \text{spe} = \frac{B - D + C}{A + B - D} = \frac{B - D + C}{392413}$$

En prenant le seuil 3 de la méthode 2, on a :

Sens=0.7 et Spe=0.71 \rightarrow 1-spe=0.28

\rightarrow c'est la courbe du bas

\rightarrow La courbe du haut est la première méthode et celle du bas la seconde



Matrice de confusion pour un problème multi-classe :

Classe trouvée →

Classe réelle ↓

	1	2	3
1	Nb d'exemples 1 étiquetés 1	Nb d'exemples 1 étiquetés 2	Nb d'exemples 1 étiquetés 3
2	Nb d'exemples 2 étiquetés 1	Nb d'exemples 2 étiquetés 2	Nb d'exemples 2 étiquetés 3
3	Nb d'exemples 3 étiquetés 1	Nb d'exemples 3 étiquetés 2	Nb d'exemples 3 étiquetés 3

Exercice

On donne la matrice de confusion ci-dessous :

Classe trouvée →

Classe réelle ↓

	1	2	3
1	90	6	5
2	20	70	4
3	2	1	104

Quel est le nombre d'exemples de la base de test ?

Que représente le chiffre 20 ?

Que représente le chiffre 104 ?

Quel est le taux de bonne reconnaissance ?

Exercice

On donne la matrice de confusion ci-dessous :

Classe trouvée →

Classe réelle ↓

	1	2	3
1	90	6	5
2	20	70	4
3	2	1	104

Quel est le nombre d'exemples de la base de test ?

La somme des éléments de la matrice = 302

Que représente le chiffre 20 ?

Le nombre d'exemples appartenant à la classe 2 étiquetés 1

Que représente le chiffre 104 ?

Le nombre d'exemples appartenant à la classe 3 étiquetés 3

Quel est le taux de bonne reconnaissance ?

La somme des éléments de la diagonale divisée par la somme des éléments

$264/302 \rightarrow 87,4 \%$

On peut réduire la matrice de confusion dès lors qu'on se focalise sur une classe.

Par exemple, pour la classe 1 :

	1	autre
1	Nb d'exemples 1 étiquetés 1	Nb d'exemples 1 étiquetés autre
autre	Nb d'exemples autre étiquetés 1	Nb d'exemples autre étiquetés autre

On peut alors redéfinir pour chaque classe le rappel, la précision, la sensibilité et la spécificité

Puis

- les mesures micro (moyenne pondérée par l'effectif de chaque classe)
- Les mesures macro (moyenne)

Exercice

Pour le même exemple que précédemment, donner la micro-précision et la macro-précision:

Classe trouvée →

Classe réelle ↓

	1	2	3
1	90	6	5
2	20	70	4
3	2	1	104

Exercice

Pour le même exemple que précédemment, donner la micro-précision et la macro-précision:

	1	2	3	Nb_ex
1	90	6	5	101
2	20	70	4	94
3	2	1	104	107
Précision	90/112	70/77	104/113	

Micro-précision = $(90/112 * 101 + 70/77 * 94 + 104/113 * 107) / (101 + 94 + 107) = 87,779\%$

Macro-précision = $(90/112 + 70/77 + 104/113) / 3 = 87,7672\%$

Algorithme des kppv

KPPV : K plus proches voisins

Il s'agit d'une **méthode supervisée et discriminative**

Données de départ

Ensemble d'exemples de dimensions n étiquetés $y \in [1, \dots, C]$

But

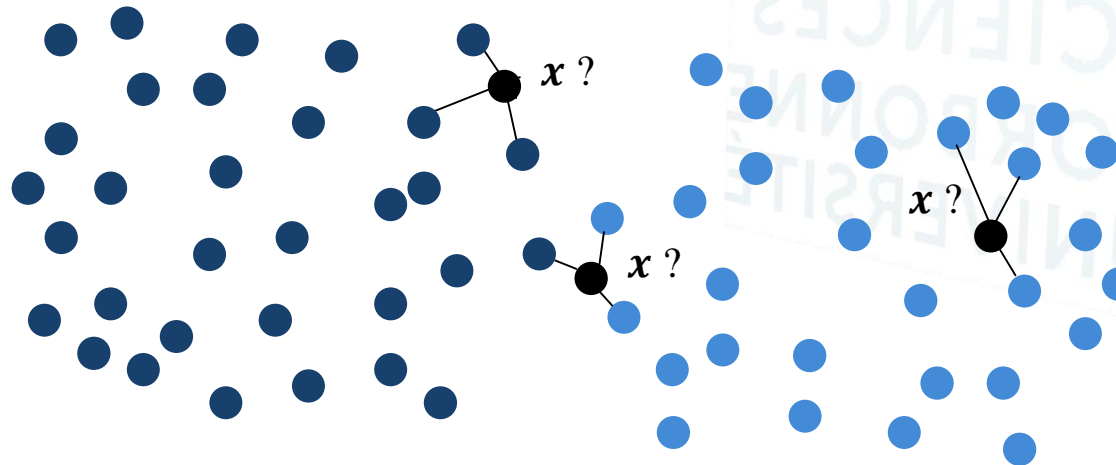
Trouver la classe d'un nouvel exemple x

Méthode

- Calculer la distance entre x et tous les exemples de la base de référence
- Déterminer les K exemples les plus proches.
- Affecter à x la classe majoritaire parmi les K plus proches voisins

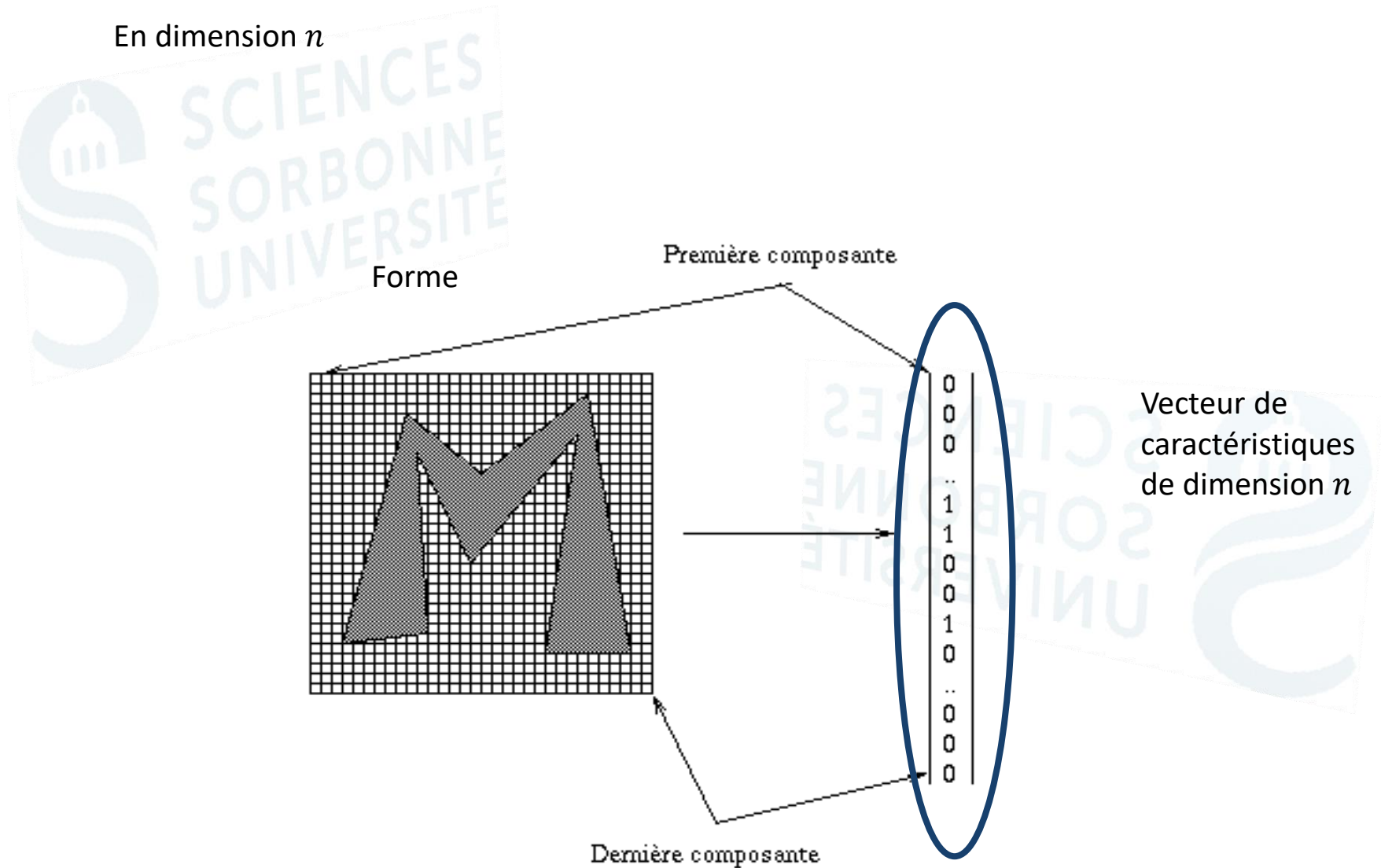
Exemple

En dimension 2, chaque exemple est caractérisé par un vecteur de dimension 2 et est donc représenté dans le plan :

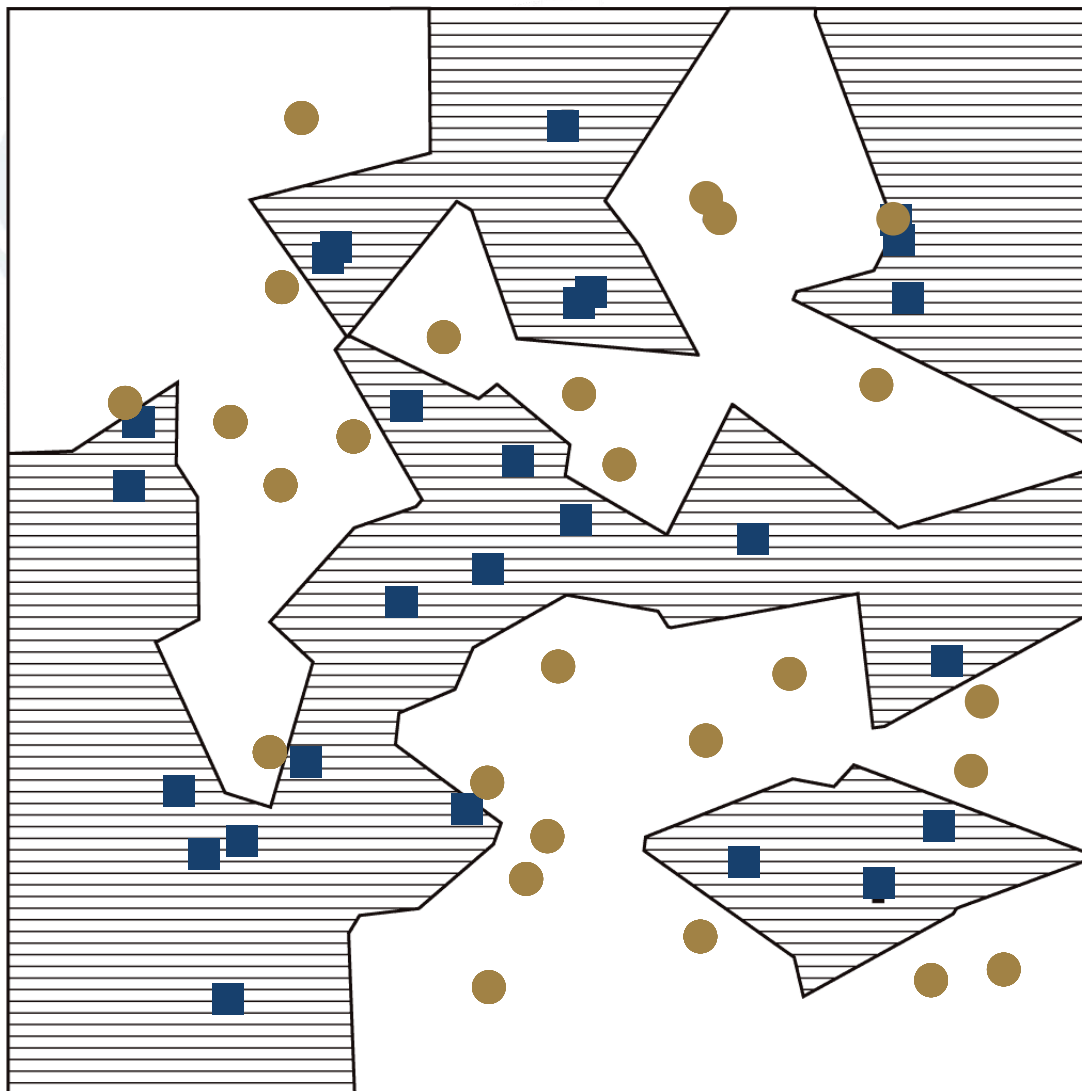


Exemple

En dimension n



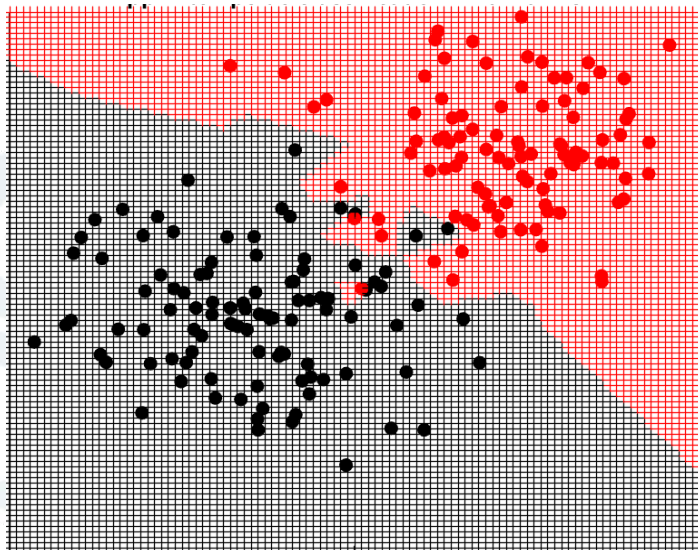
Signification géométrique 1ppv



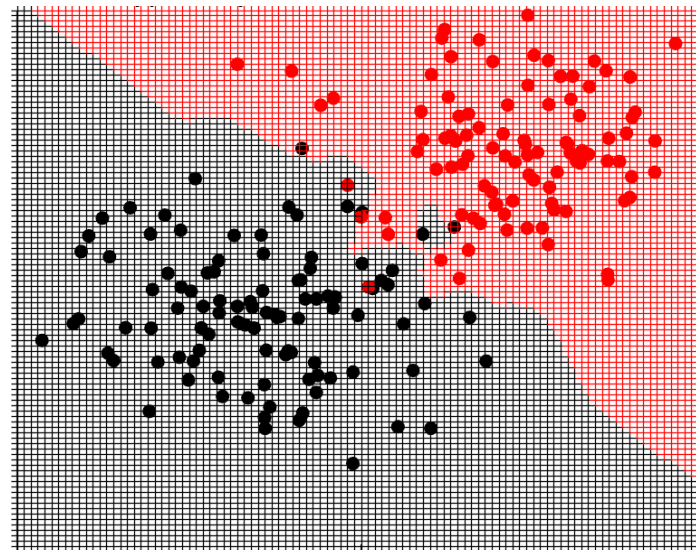
Les classes sont définies par la réunion des domaines d'influence des exemples de références

La résolution spatiale des frontières est liée au nombre d'exemples et à leur densité

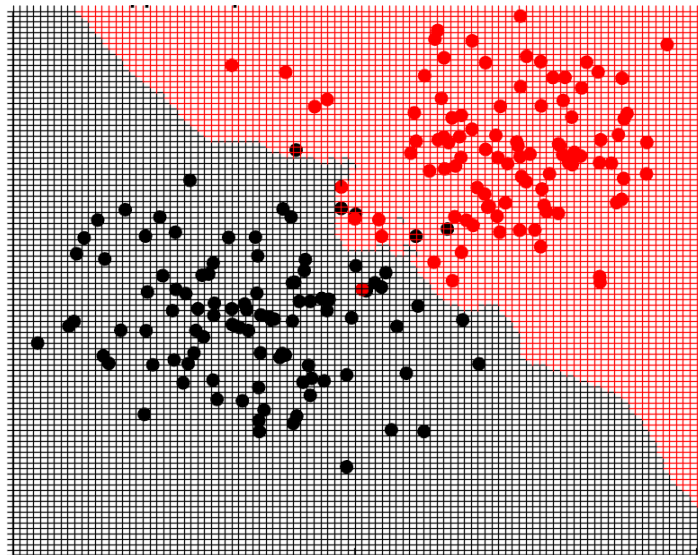
k=1



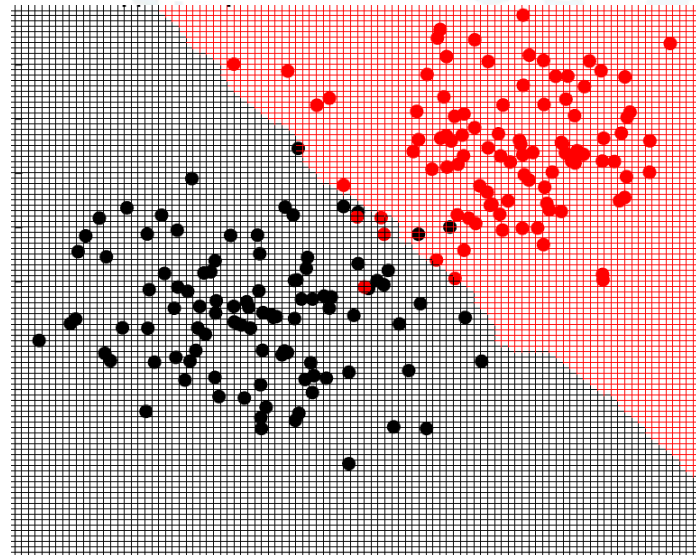
k=3



k=5



k=11



Dilemme biais/variance

k faible

- ➔ Bonne résolution des frontières entre classe
- ➔ Très sensible aux échantillons de la base de référence
- ➔ Petit biais Grande variance

k grand

- ➔ Mauvaise résolution des frontières entre classe : lissage des frontières
- ➔ Peu sensible aux échantillons de la base de référence
- ➔ Grand biais Faible variance

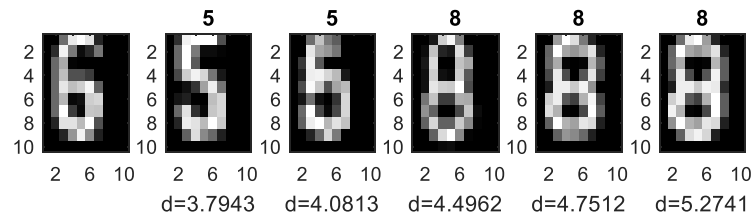
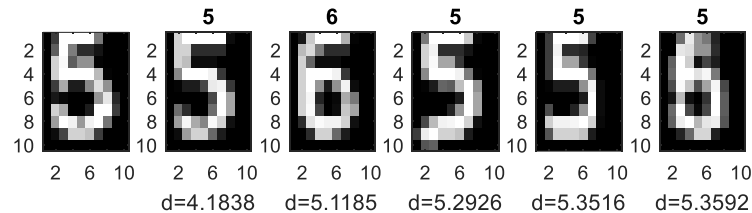
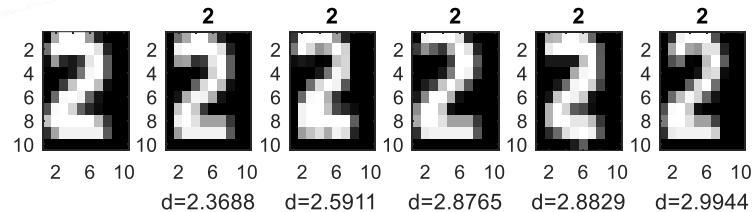
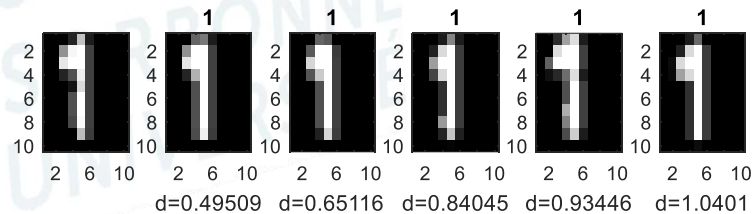
Comment choisir k ?

3 bases de données

- ➔ Base de référence où sont stockés les exemples
- ➔ Base de validation utilisée pour optimiser k
- ➔ Base de test pour évaluer les performances

Exemple en reconnaissance de caractères

- 800 exemples dans la base de références
- Chaque exemple est de dimension 100 (codage rétinien 10x10)



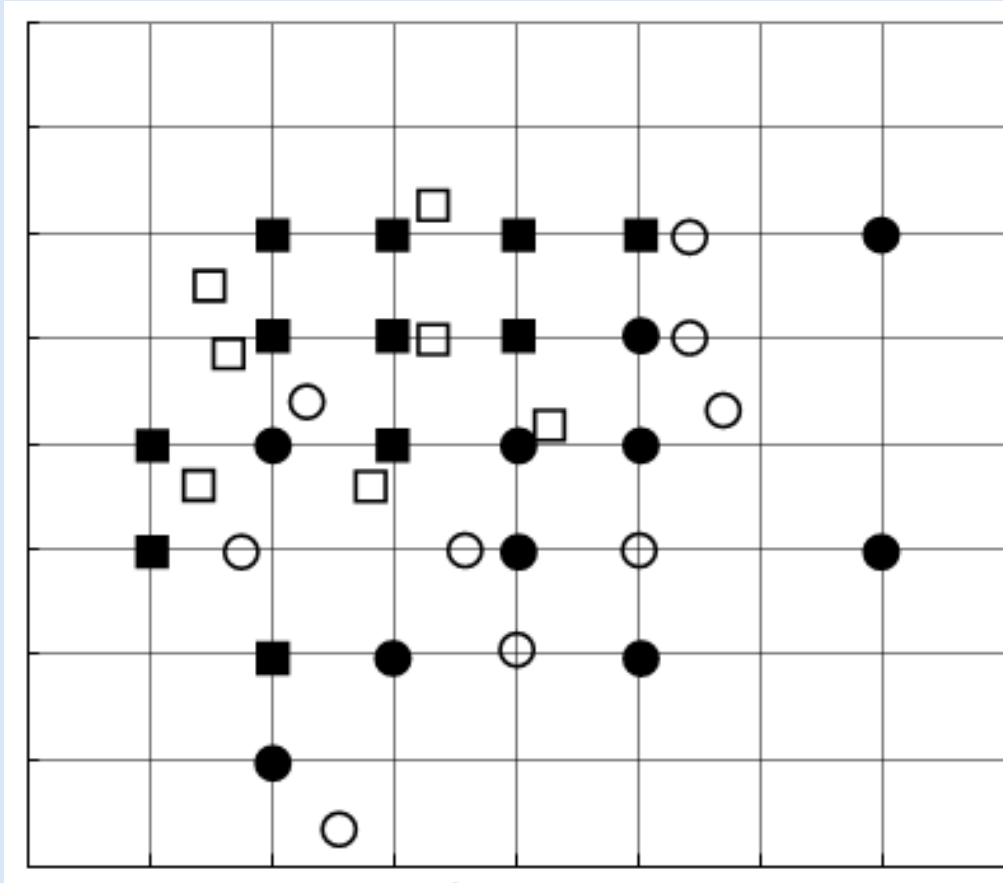
Avantages :

- Pas d'hypothèses
- Simple à mettre en œuvre
- Incrémental

Inconvénients :

- Quantité de calculs quasi-proportionnelle au nombre d'exemples
- Pas d'extraction d'information utile

Exercice

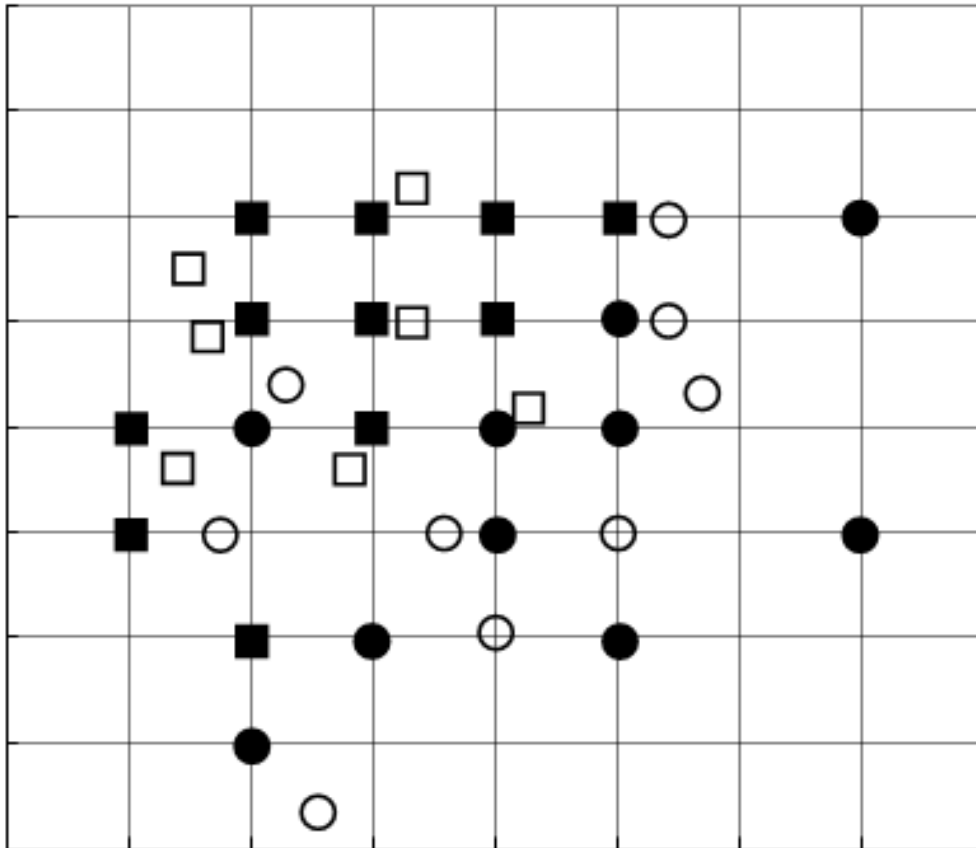


Noir : base de référence

Blanc : base de test

Donner la matrice de confusion avec
l'algorithme du 1ppv

Exercice



Noir : base de référence

Blanc : base de test

	□	○
□	6	1
○	2	7

Algorithme des KPPV

Approche 'nearest mean'

Accélération des k-PPV

2 solutions

Réduction de la dimension de chaque exemple

- ACP
- LDA

Réduction de taille de la base de référence

- On ne représente plus chaque classe que par sa moyenne

Dilemme robustesse/accélération



Algorithme des KPPV

Approche 'nearest mean'

Représentation de chaque classe par sa moyenne μ_c

→ distances euclidiennes $d_E(x, \mu_c)$ entre l'exemple à classe x et les centres μ_c

$$d_E(x, c) = (x - \mu_c)^T (x - \mu_c)$$

→ L'exemple est classé à la classe dont le centre est plus proche au sens de la distance euclidienne

Représentation de chaque classe par sa moyenne μ_k et sa matrice de covariance Σ_c

→ distances de Mahalanobis $d_M(x, c)$ entre l'exemple à classe x et les centres μ_c

$$d_M(x, c) = (x - \mu_c)^T \Sigma_c^{-1} (x - \mu_c)$$

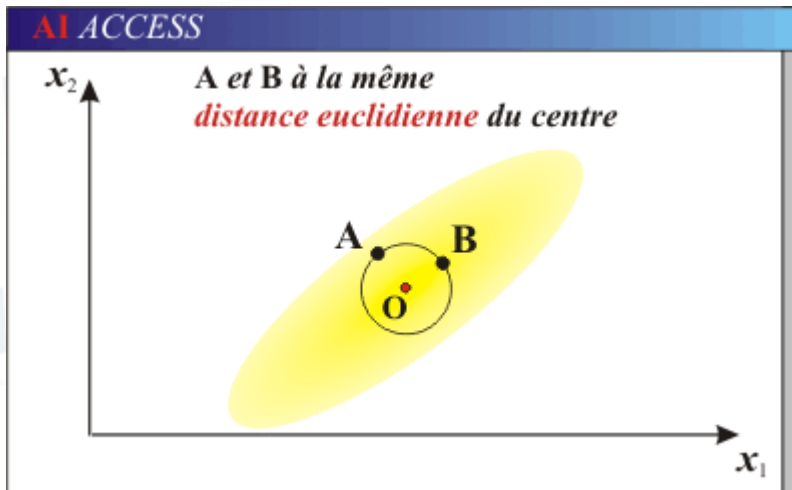
→ L'exemple est classé à la classe dont le centre est plus proche au sens de la distance de Mahalanobis

Rq : Si Σ =Identité, on retrouve la distance euclidienne

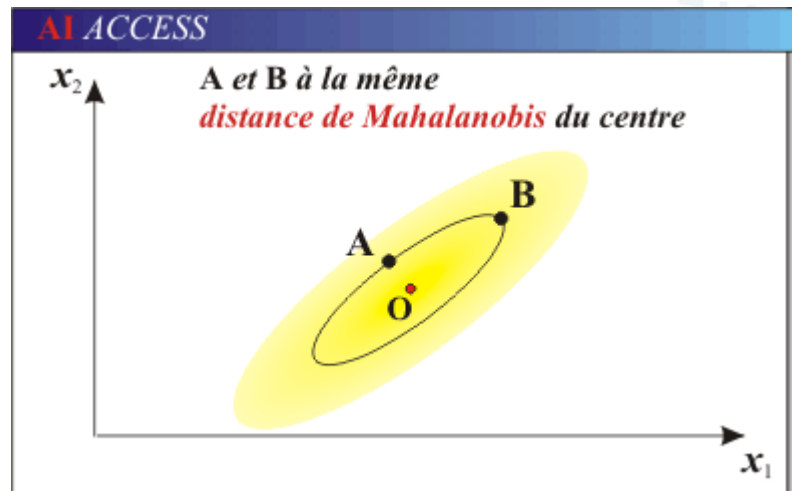
Algorithme des KPPV

Approche 'nearest mean'

Comparaison – distance euclidienne – distance de Mahalanobis



Les deux points A et B sont à la même distance euclidienne de O
→ Pas logique

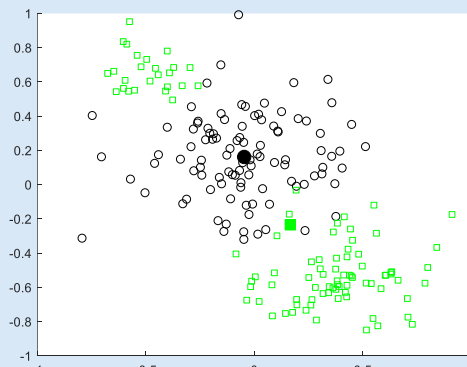
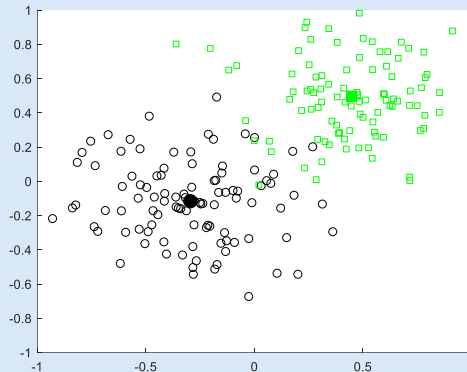
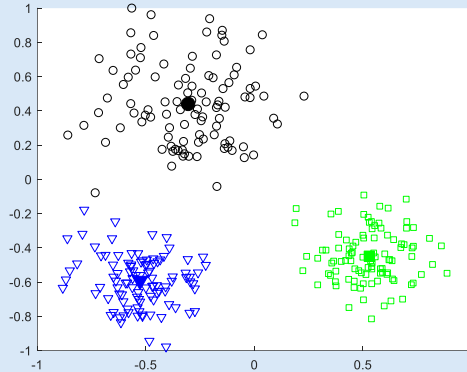


Les deux points A et B sont à la même distance de Mahalanobis de O

Algorithme des KPPV

Approche 'nearest mean'

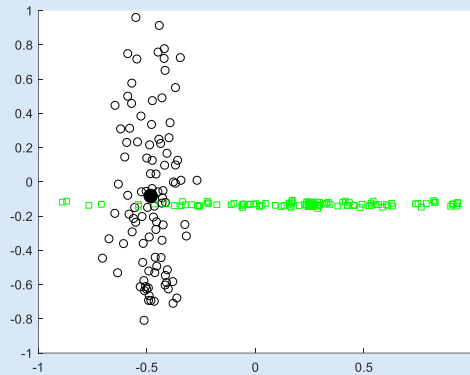
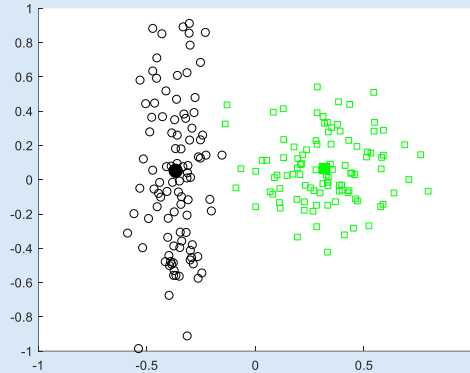
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

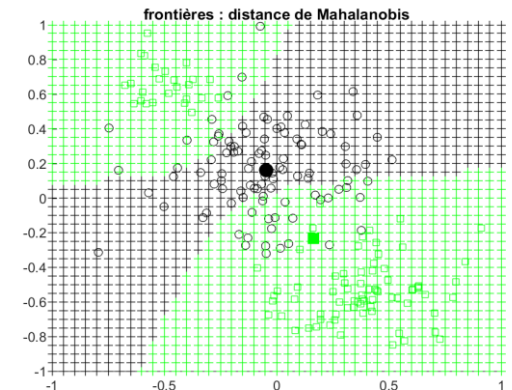
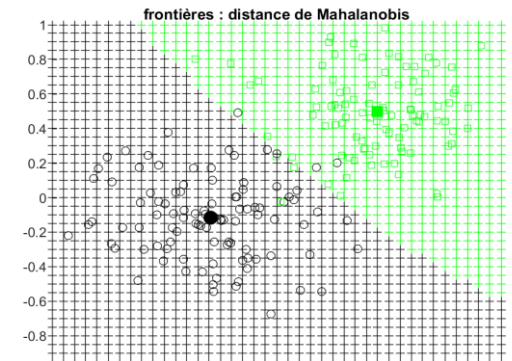
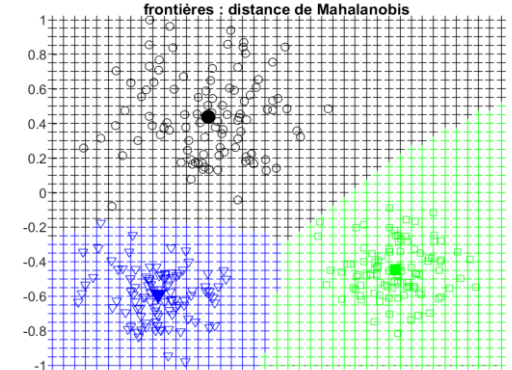
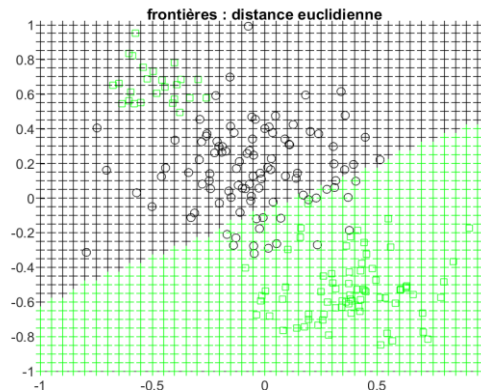
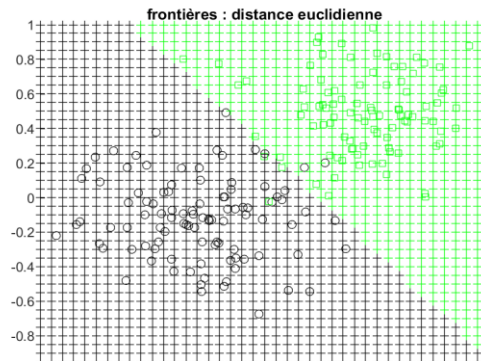
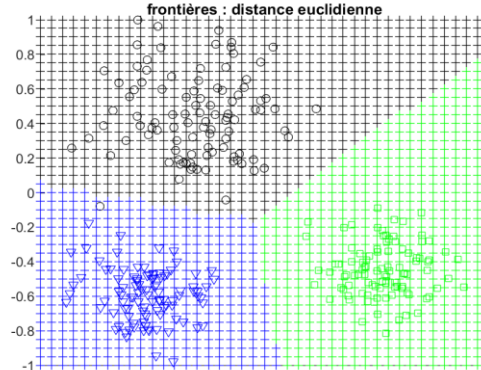
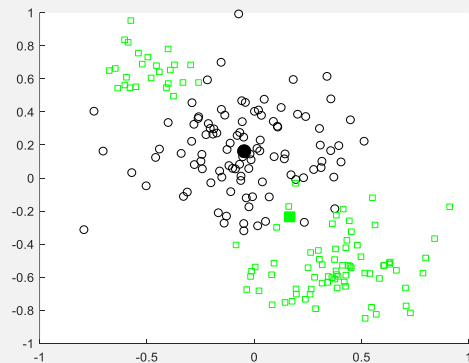
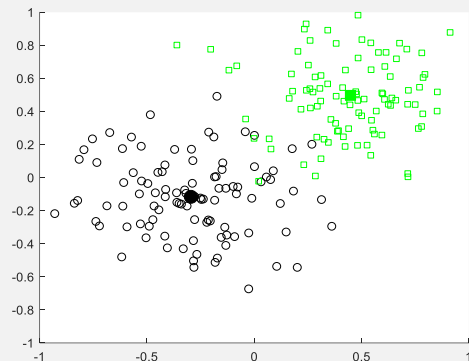
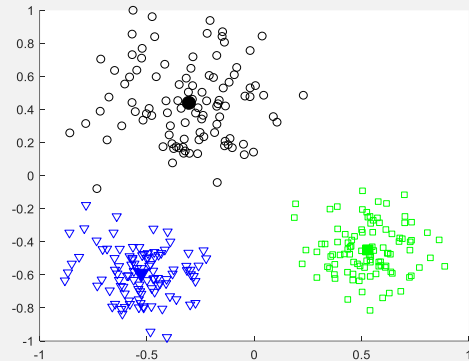
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

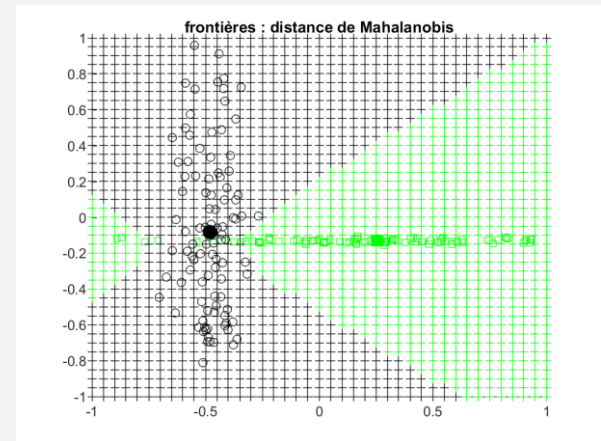
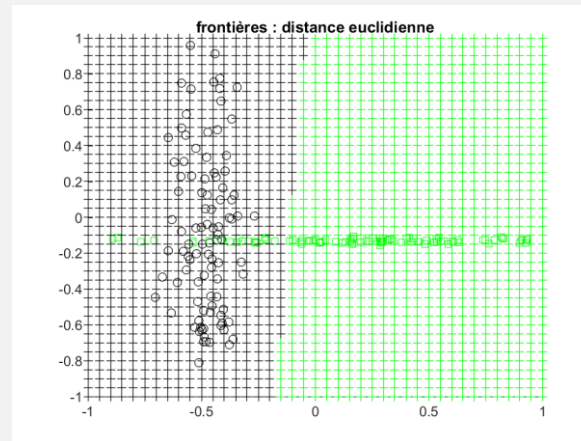
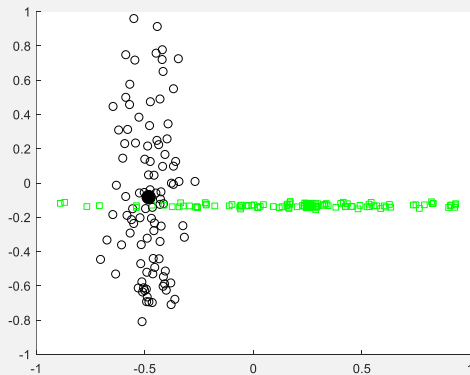
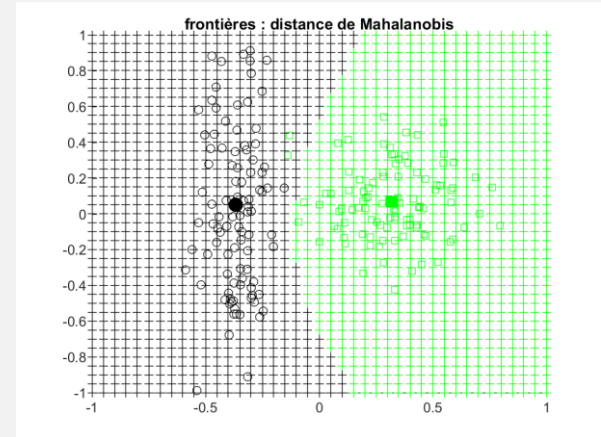
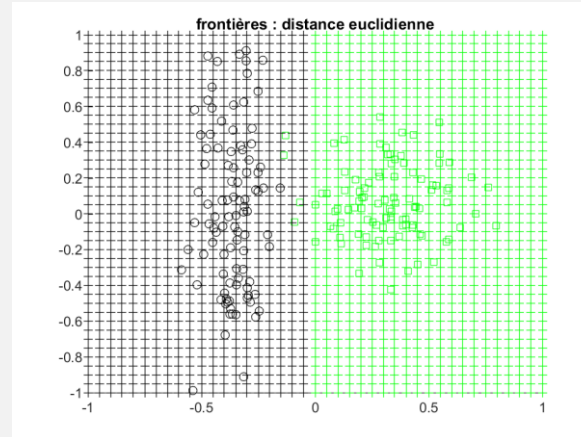
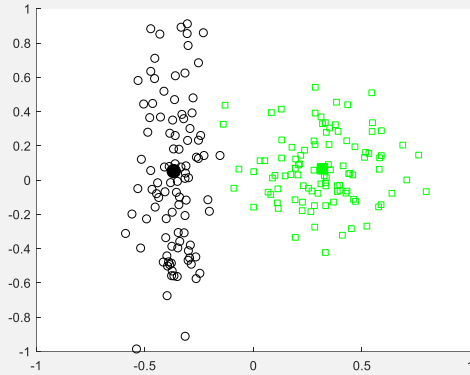
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

Exercice:

Supposons que l'on ait un problème à 3 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe :

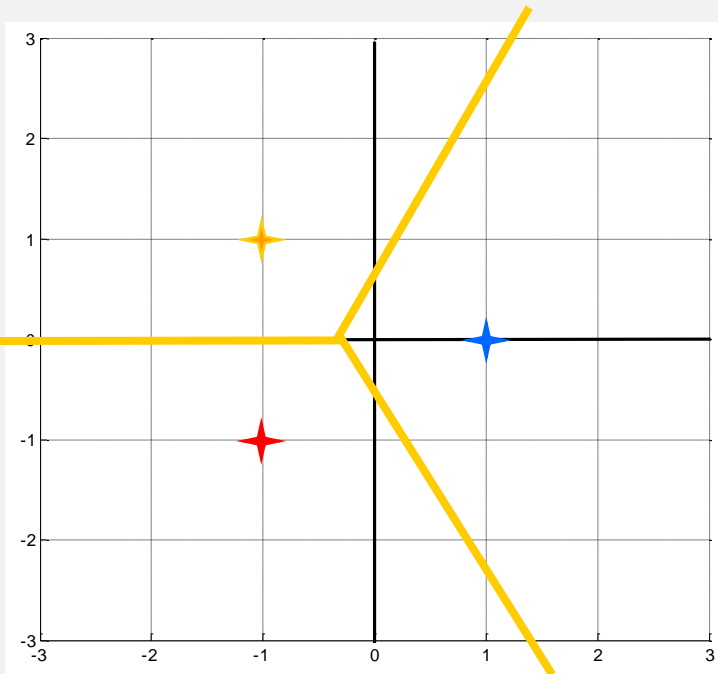
$$\begin{aligned} m1 &= \begin{pmatrix} -1 \\ 1 \end{pmatrix} & m2 &= \begin{pmatrix} -1 \\ -1 \end{pmatrix} & m3 &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ \Sigma1 &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \Sigma2 &= \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} & \Sigma3 &= \begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix} \end{aligned}$$

1. Tracer les frontières entre les classes en utilisant l'algorithme *nearest-mean* et la distance euclidienne.
2. Tracer les frontières approximatives entre les classes en utilisant l'algorithme *nearest-mean* et la distance de Mahalanobis .

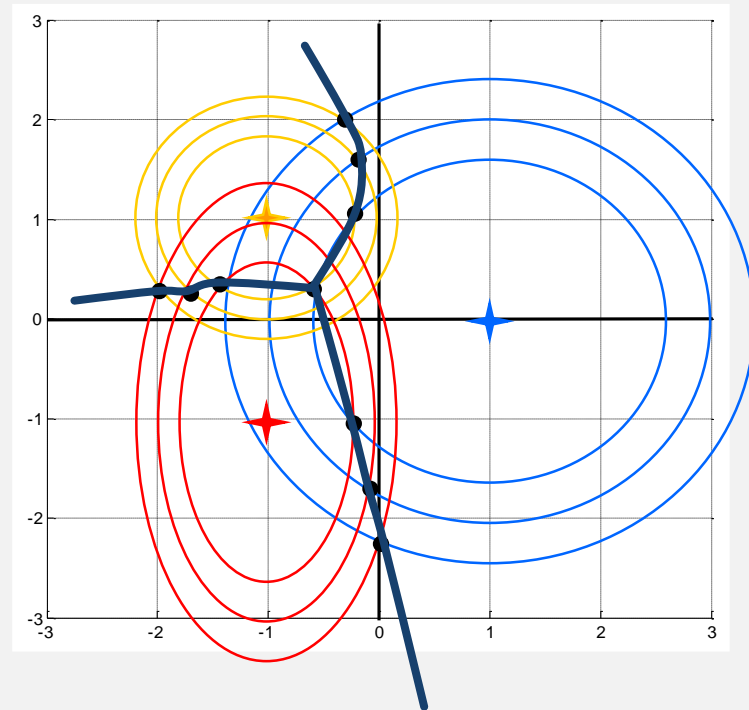
Algorithme des KPPV

Approche 'nearest mean'

Distance euclidienne



Distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

Exercice:

Supposons que l'on ait un problème à 2 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe:

$$m1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad m2 = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \quad \Sigma1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Sigma2 = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$$

3. Considérons le point $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Estimer la distance de ce point à chaque classe en utilisant l'approche nearest mean et la distance euclidienne puis la distance de Mahalanobis

Algorithme des KPPV

Approche 'nearest mean'

Exercice:

Supposons que l'on ait un problème à 2 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe:

$$m1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad m2 = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \quad \Sigma1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Sigma2 = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$$

3. Considérons le point $x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Estimer la distance de ce point à chaque classe en utilisant l'approche nearest mean et la distance euclidienne puis la distance de Mahalanobis

Avec la distance euclidienne,

$$d1 = (x - m1)^T (x - m1) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2$$

$$\text{et } d2 = (x - m2)^T (x - m2) = \begin{pmatrix} -2 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = 5$$

➔ Le point appartiendrait à la classe 1

Avec la distance de Mahalanobis,

$$d1 = (x - m1)^T \Sigma1^{-1} (x - m1) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2$$

$$\text{et } d2 = \begin{pmatrix} -2 & 1 \end{pmatrix} \begin{pmatrix} 0.2 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.4 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = 1,8$$

➔ Le point appartiendrait à la classe 2