

IFT3395/6390

Fondements de l'apprentissage machine

Formalisation des problèmes d'apprentissage

Méthodes de type histogramme

Malédiction de la dimensionalité

Professeur: Pascal Vincent

Au programme aujourd'hui

- ◆ Petit rappel de terminologie.
- ◆ Ex. de problème de classification, régression, estimation de densité
- ◆ Méthodes de type **histogramme**, illustrées pour classification, régression, estimation de densité.
- ◆ **Malédiction de la dimensionalité.**
- ◆ Formalisation du problème de l'apprentissage. Notion de capacité.

D_n

Ensemble de données
d'entraînement (*training set*)

Taille de
l'ensemble,
nombre
d'exemples:

n

entrées:



cibles:

"cheval"



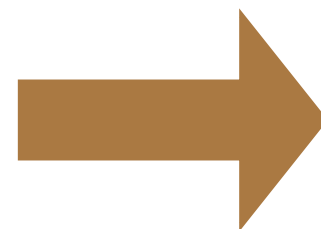
"chat"

etc...



"cheval"

prétraitement,
extraction de
caractéristiques



preprocessing,
feature
extraction

X_1

entrées: X
(vecteur de traits caractéristiques)

(3.5, -2, ... , 127, 0, ...)

+1

Y_1

(-9.2, 32, ... , 24, 1, ...)

-1

etc...

X_n

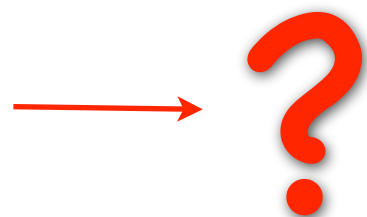
$X_{n,2}$

(6.8, 54, ... , 17, -3, ...)

+1

Y_n

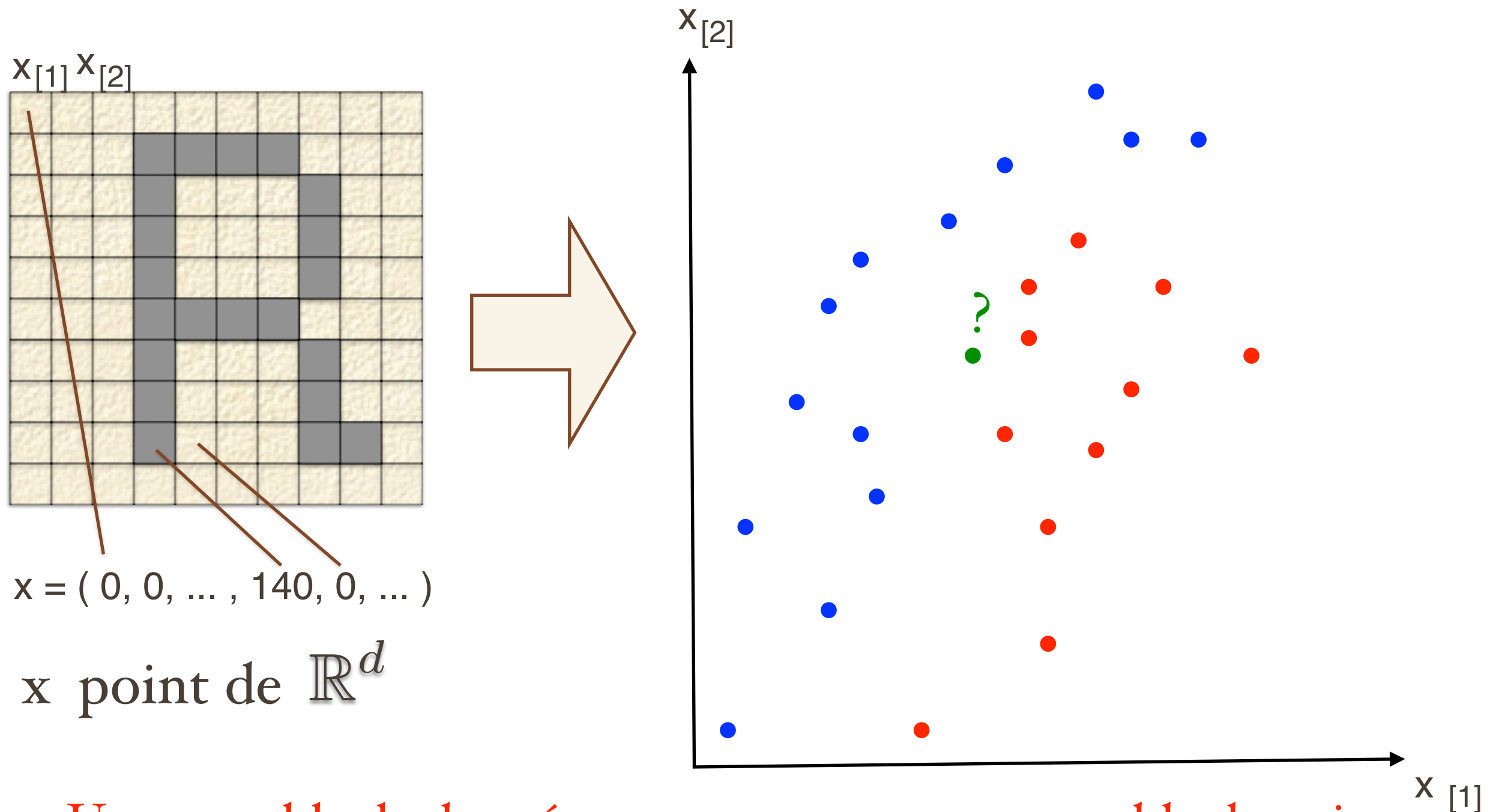
Point de test:



$x = (5.7, -27, ... , 64, 0, ...)$



Représentation des données



Un ensemble de données est vu comme un ensemble de points en haute dimension.

Terminologie de l'apprentissage supervisé

Ensemble de données d'entraînement (*training set*)

Une **entrée** est généralement représentée par un vecteur de dimension **d**.

$x \in \mathbb{R}^d$
↑
dimensionnalité de l'entrée

| | | |
|---|-------------------------------------------|----------------------------------------------|
| 1 | entrée, observation, <i>input</i> , x_1 | cible, <i>target</i> , sortie désirée, y_1 |
| 2 | entrée, observation, <i>input</i> , x_2 | cible, <i>target</i> , sortie désirée, y_2 |
| 3 | entrée, observation, <i>input</i> , x_3 | cible, <i>target</i> , sortie désirée, y_3 |
| ⋮ | <i>etc...</i> | ⋮ |
| n | entrée, observation, <i>input</i> , x_n | cible, <i>target</i> , sortie désirée, y_n |

point de test



? ? ?

taille de l'ensemble, nombre d'exemples, d'échantillons.

On cherche un algorithme qui produit une **sortie** (*output*) qui est une bonne prédiction de la cible.
Cet algorithme trouve une bonne fonction $x \rightarrow y$

Problème d'apprentissage

- Données

- $D_n = (Z_1, Z_2, \dots, Z_n)$ générées par la “nature”
pour l'apprentissage supervisé $Z_i = (X_i, Y_i)$

- **IID**: “independent and identically distributed”

- tirés de la même distribution INCONNUE $p(Z)$
 - de manière indépendante

signifie que l'ordre des exemples ne contient pas d'information
=> on devrait pouvoir les mélanger sans que ça ait d'influence!

(Attention: IID n'est pas valide pour tous les types de données!)

Problème d'apprentissage

- Les trois problèmes considérés

- **classification**: $Z = (X, Y) \in \mathbb{R}^d \times \{-1, 1\}$ ou $\{0, 1\}$ classification binaire
ou bien $\{1, \dots, m\}$ ou $\{0, 1, \dots, m-1\}$ classification multiclasse
- **régression**: $Z = (X, Y) \in \mathbb{R}^d \times \mathbb{R}$
- **estimation de densité**: $Z \in \mathbb{R}^d$

- Ensemble de fonctions F (solutions possibles), $f \in F$:

- **classification**: $f : \mathbb{R}^d \rightarrow \{-1, 1\}$ ou $\{0, 1\}$ classification binaire
ou bien $\{1, \dots, m\}$ ou $\{0, 1, \dots, m-1\}$ classification multiclasse
- **régression**: $f : \mathbb{R}^d \rightarrow \mathbb{R}$
- **estimation de densité**: F contient des fonctions de densité

Les types de problèmes ou tâches classiques en apprentissage

| | supervisé | supervisé | non-supervisé |
|---------------------------------------------------------------------|----------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| | Classification | Régression | Estimation de densité |
| Signification de la cible y | indique une classe parmi m classes. | une valeur réelle à prédire. | pas de cible y ! |
| Domaine de y | $y \in \{-1, 1\}$ ou $y \in \{1, \dots, m\}$ ou $y \in \{0, 1, \dots, m-1\}$ | $y \in \mathbb{R}$ | pas de cible y ! |
| Ce que $f(x)$ vise à prédire | la classe de x (la classe la plus probablement associée à x) | la valeur espérée de y (le y "moyen") correspondant à x . $E[Y \mid X=x]$ | la densité $p(x)$ (l'observation x est-elle fort ou peu probable?) |
| Fonction de perte (ou coût) que l'on veut habituellement minimiser. | l'erreur de classification: $L((x, y), f) = I_{\{f(x) \neq y\}}$ | l'erreur quadratique: $L((x, y), f) = (f(x) - y)^2$ | la log-vraisemblance négative: $L(x, f) = -\log f(x)$ |

Ex. de problème de classification

Iris flower data set

From Wikipedia, the free encyclopedia

The **Iris flower data set** or **Fisher's Iris data set** is a multivariate data set introduced by Sir Ronald Aylmer Fisher (1936) as an example of discriminant analysis.^[1] It is sometimes called **Anderson's Iris data set** because Edgar Anderson collected the data to quantify the geographic variation of *Iris* flowers in the Gaspé Peninsula.^[2]

The dataset consists of 50 samples from each of three species of *Iris* flowers (*Iris setosa*, *Iris virginica* and *Iris versicolor*). Four features were measured from each sample, they are the length and the width of sepal and petal, in centimeters. Based on the combination of the four features, Fisher developed a linear discriminant model to distinguish the species from each other. It is used as a typical test for many other classification techniques.

$$\mathbf{x} \in \mathbb{R}^4$$

Fisher's Iris Data

Y

| Sepal Length | Sepal Width | Petal Length | Petal Width | Species |
|--------------|-------------|--------------|-------------|---------------|
| 5.1 | 3.5 | 1.4 | 0.2 | <i>setosa</i> |
| 4.9 | 3.0 | 1.4 | 0.2 | <i>setosa</i> |
| 4.7 | 3.2 | 1.3 | 0.2 | <i>setosa</i> |
| 4.6 | 3.1 | 1.5 | 0.2 | <i>setosa</i> |
| 5.0 | 3.6 | 1.4 | 0.2 | <i>setosa</i> |
| 5.4 | 3.9 | 1.7 | 0.4 | <i>setosa</i> |
| 4.6 | 3.4 | 1.4 | 0.3 | <i>setosa</i> |

etc...

| | | | | |
|-----|-----|-----|-----|-------------------|
| 5.7 | 2.8 | 4.5 | 1.3 | <i>versicolor</i> |
| 6.3 | 3.3 | 4.7 | 1.6 | <i>versicolor</i> |
| 4.9 | 2.4 | 3.3 | 1.0 | <i>versicolor</i> |
| 6.6 | 2.9 | 4.6 | 1.3 | <i>versicolor</i> |
| 5.2 | 2.7 | 3.9 | 1.4 | <i>versicolor</i> |
| 5.0 | 2.0 | 3.5 | 1.0 | <i>versicolor</i> |

etc...

| | | | | |
|-----|-----|-----|-----|------------------|
| 7.7 | 3.0 | 6.1 | 2.3 | <i>virginica</i> |
| 6.3 | 3.4 | 5.6 | 2.4 | <i>virginica</i> |
| 6.4 | 3.1 | 5.5 | 1.8 | <i>virginica</i> |
| 6.0 | 3.0 | 4.8 | 1.8 | <i>virginica</i> |
| 6.9 | 3.1 | 5.4 | 2.1 | <i>virginica</i> |

etc...

n=150



Iris setosa

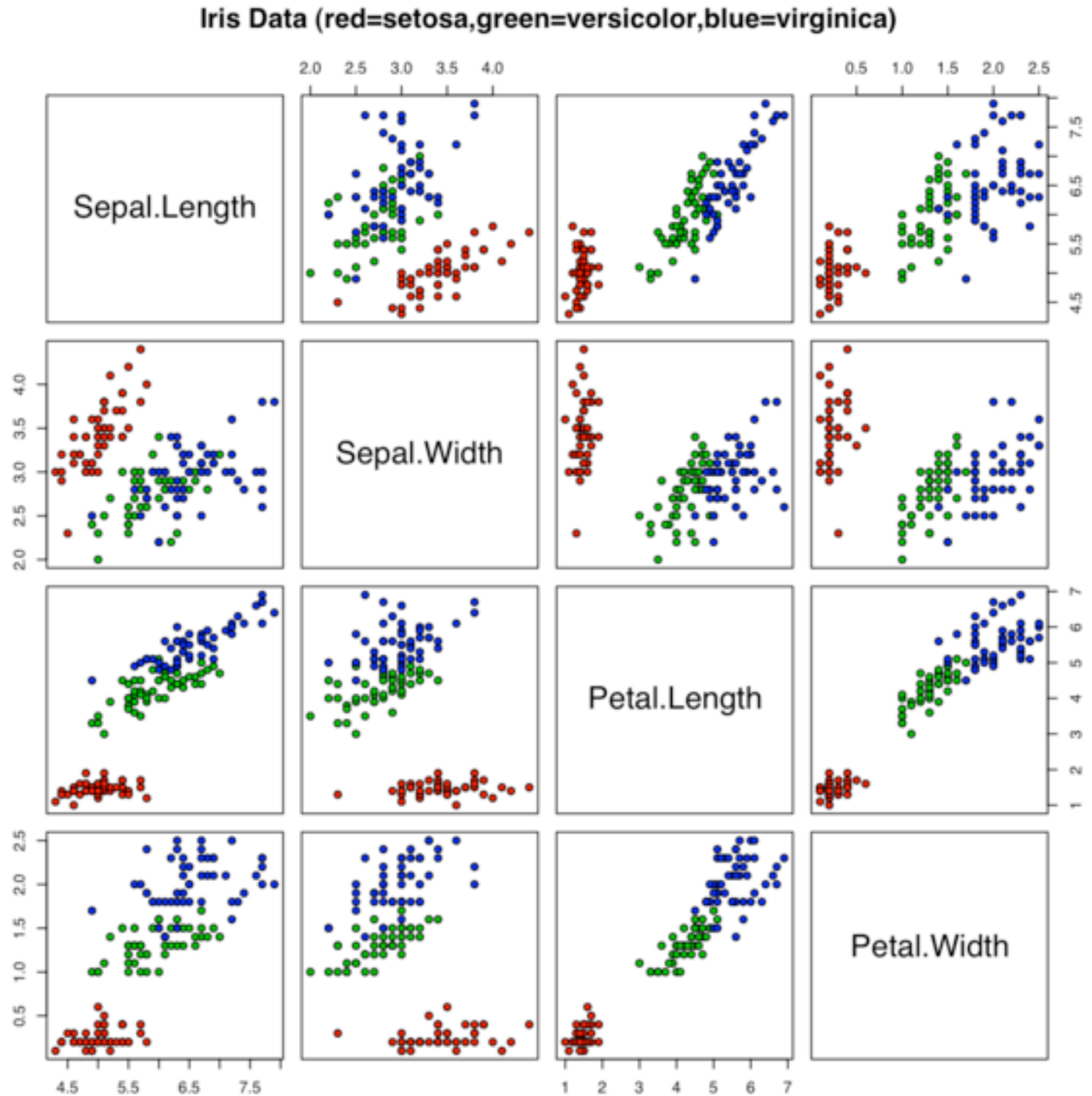


Iris versicolor



Iris virginica

La classe en
fonction de 2 traits
caractéristiques



Un même ensemble de donnée peut permettre de définir plusieurs tâches différentes!

- ◆ **Classification:** prédire l'espèce d'iris sachant les dimensions du pétale et sépale.
- ◆ **ex régression:**
prédire les dimensions du pétale sachant les dimensions du sépale et la classe d'iris
- ◆ **ex estimation de densité de probabilité:**
je mesure les dimensions de pétale et sépale d'une fleur donnée. Ressemblent-elles aux dimensions typiques d'une iris setosa?

Etapes d'un projet utilisant des algorithmes d'apprentissage

- Le **problème** à résoudre peut-il être **reformulé** sous la forme d'une des tâches standard de l'apprentissage?
(classification, régression, estimation de densité, partitionnement, réduction de dimensionalité, ...)
Quel coût veut-on réellement optimiser dans ce problème?
- **Examiner les données** dont on peut disposer pour l'entraînement/test. Y a-t-il suffisamment d'exemples?, bien comprendre leur format, leur sémantique.
- Concevoir et coder les étapes de **prétraitement des données**. Le but est de les transformer dans une forme appropriée pour les algos d'apprentissage qu'on va utiliser.
- **Entraîner/tester et évaluer** correctement la **performance** "hors échantillon" des algorithmes considérés. (ex: découpage entraînement/test, validation croisée, ou bootstrap).

Étapes pratiques d'un projet de data-mining

- Identifier clairement le problème à résoudre
- Le formaliser comme une tâche d'apprentissage spécifique basée sur les données disponibles.
- Extraire les données et les regrouper (dans un fichier, une table).
- Prétraiter les données pour obtenir une représentation appropriée pour les algorithmes d'apprentissage envisagés.
- **Modélisation**: appliquer plusieurs algorithmes d'apprentissage sur les données.
- **Évaluation de la performance** de chaque algorithme, pour choisir la meilleure approche.
- Déployer le système opérationnel chez le client.

“data plumbing”

sélection de modèle

Méthodes de type histogramme

Les algorithmes à base de quadrillages de l'espace
(de type **histogramme**)

Une idée simple: découper
l'espace en petits cubes...

Une idée simple pour la classification

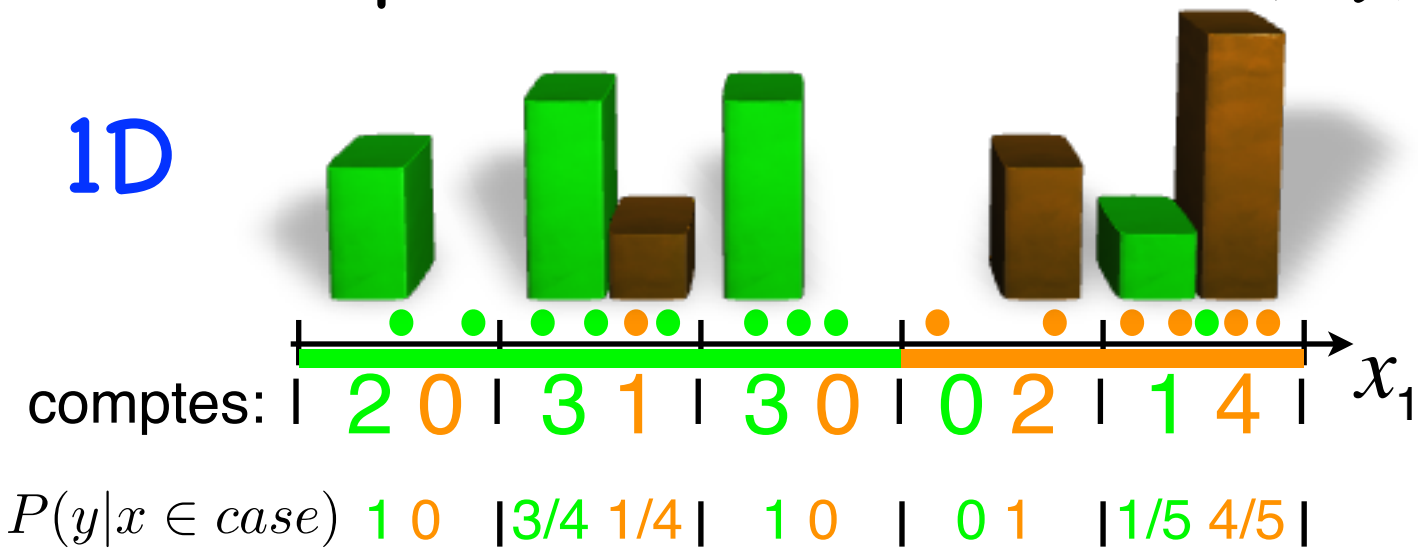
Tout algo d'apprentissage doit pouvoir effectuer une prédiction pour n'importe quel point de test de l'espace d'entrée... (ex: $x \in \mathbb{R}^d$)
Partant de là, voici une idée simple d'algorithme:

- ◆ **Quadriller l'espace!**
- ◆ **Entraînement: Compter**, pour chaque case, combien de points de chaque classe y tombent (parmi les points de l'ensemble d'apprentissage).
- ◆ **Test**: trouver la case dans laquelle tombe le point de test. Répondre la classe majoritaire tombée dans cette case.

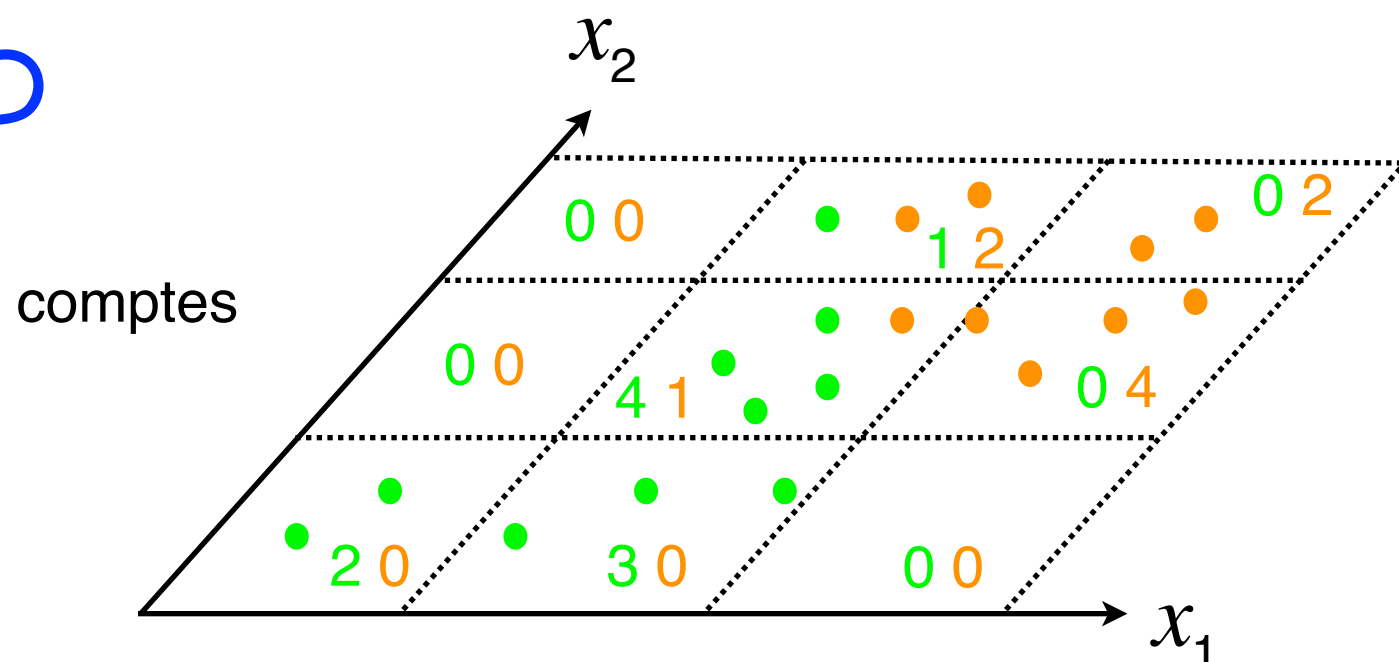
Classification

- ◆ On suppose qu'il existe un processus inconnu qui génère des paires d'observations (x, y) , ou y indique la classe (● ou ●)

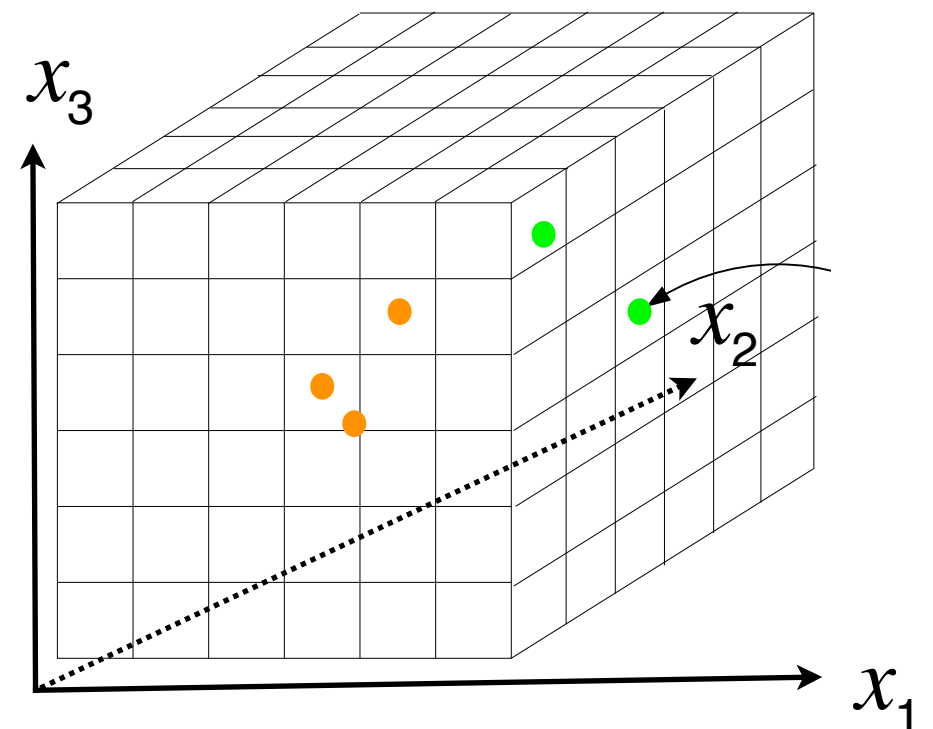
1D



2D



3D

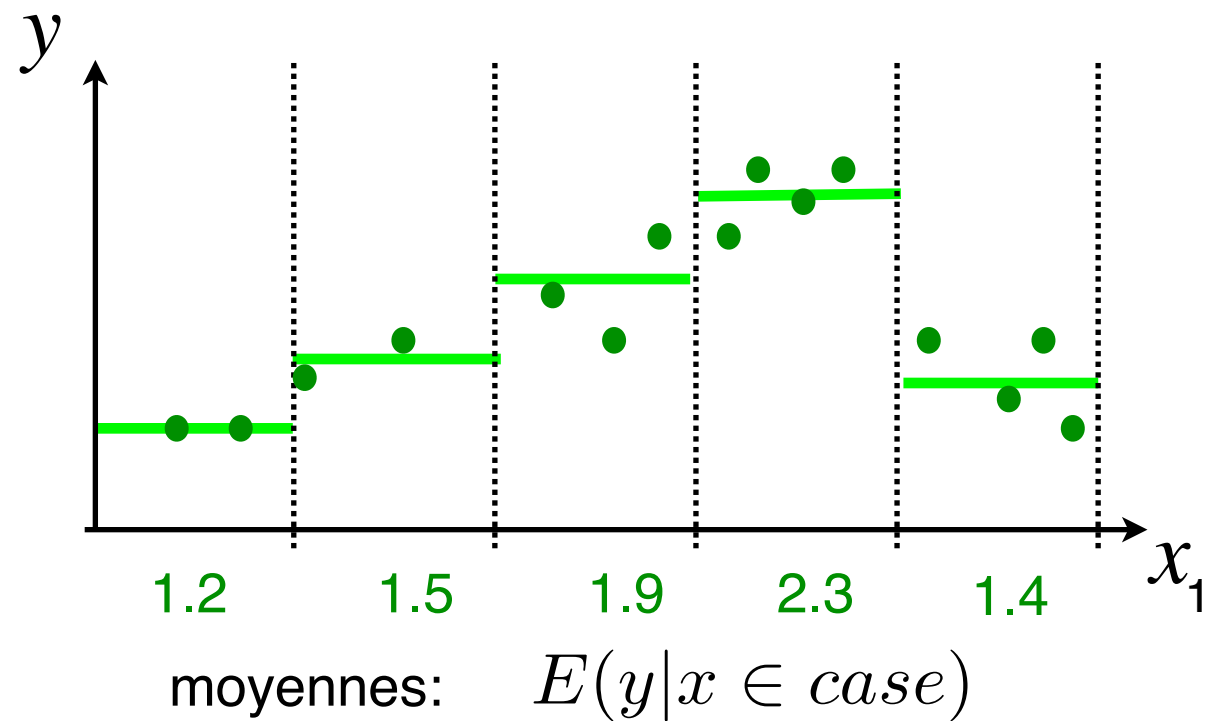


dD ...

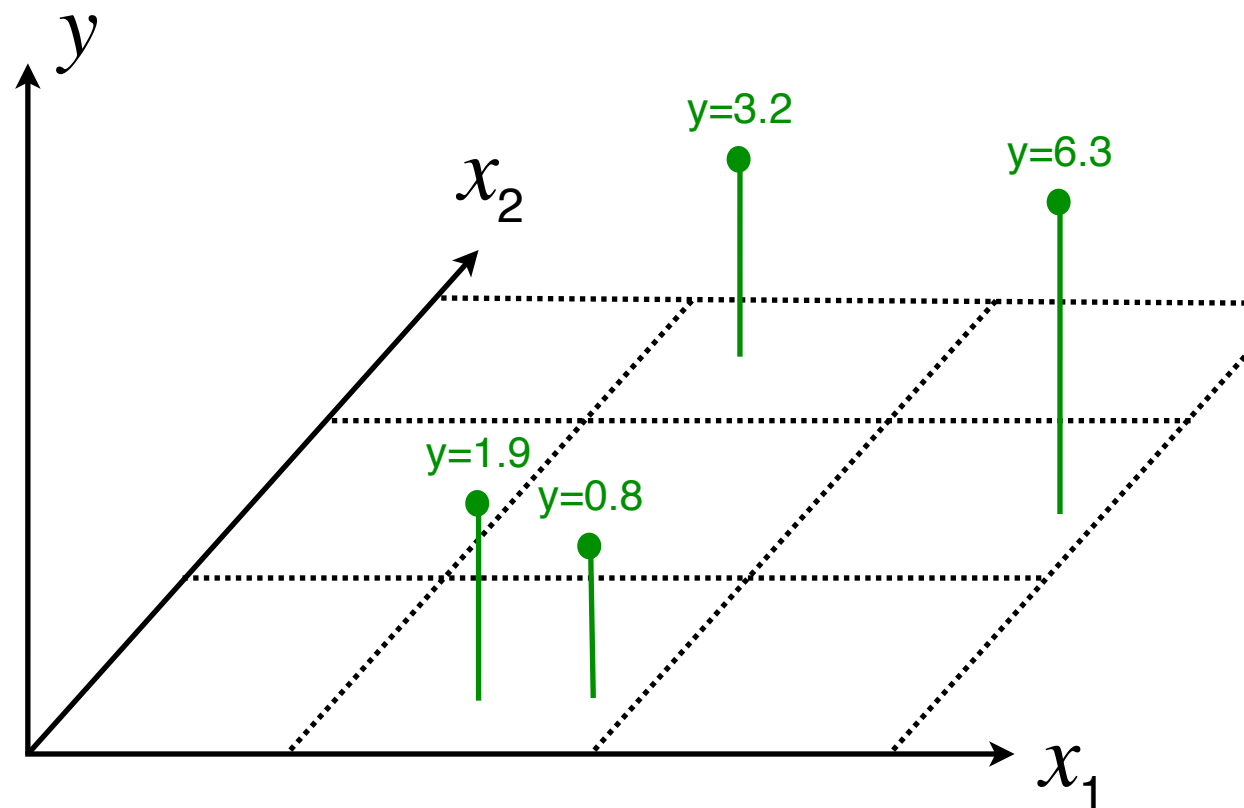
Régression

- ◆ On suppose qu'il existe un processus inconnu qui génère des paires d'observations (x,y) , avec y réel.

1D



2D



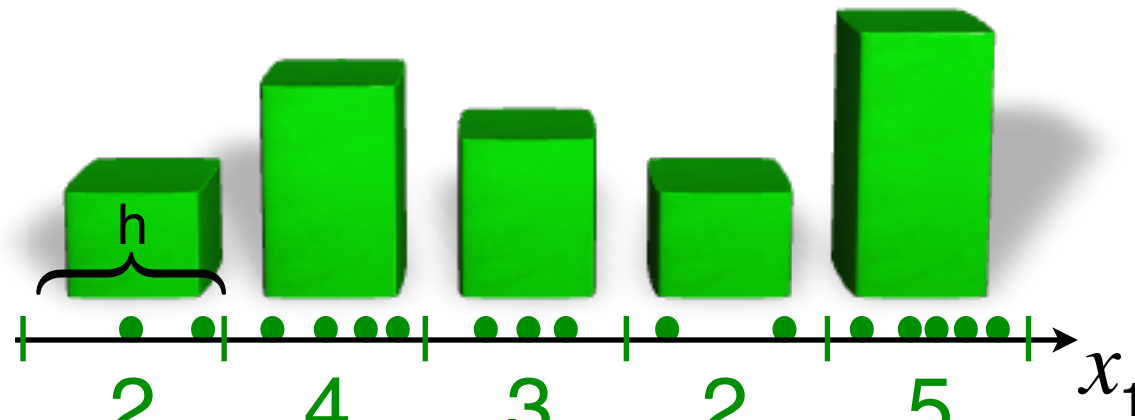
3D ...

dD ...

Estimation de densité

- ◆ On suppose qu'il existe un processus inconnu qui génère des observations x .

1D

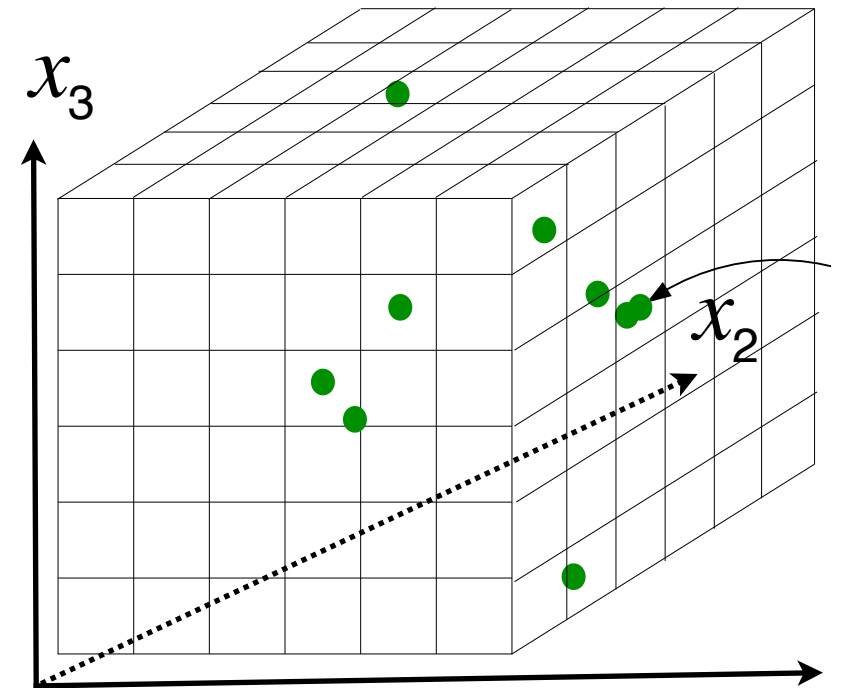


comptes:

$p(x)$ 2/16/h 4/16/h 3/16/h 2/16/h 5/16/h

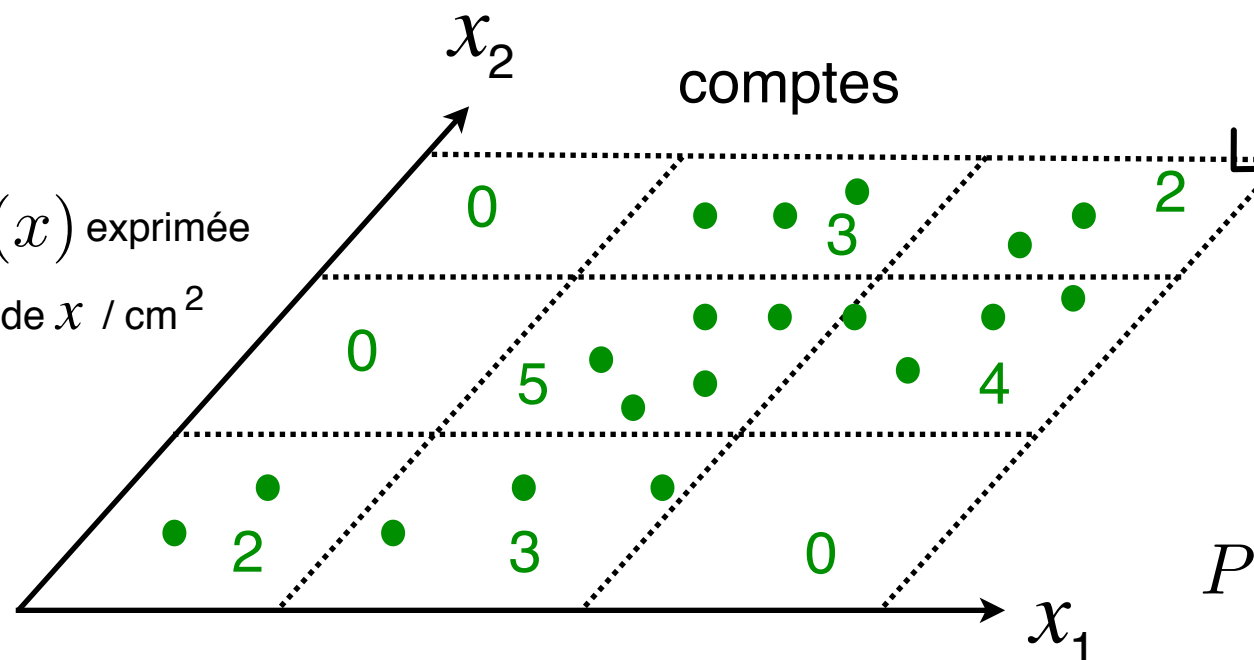
densité de probabilité:

3D



2D

Densité de probabilité $p(x)$ exprimée
par ex. en probabilité de x / cm^2



La densité de prob. doit intégrer à 1 sur tout l'espace:

$$\int p(x) dx = 1$$

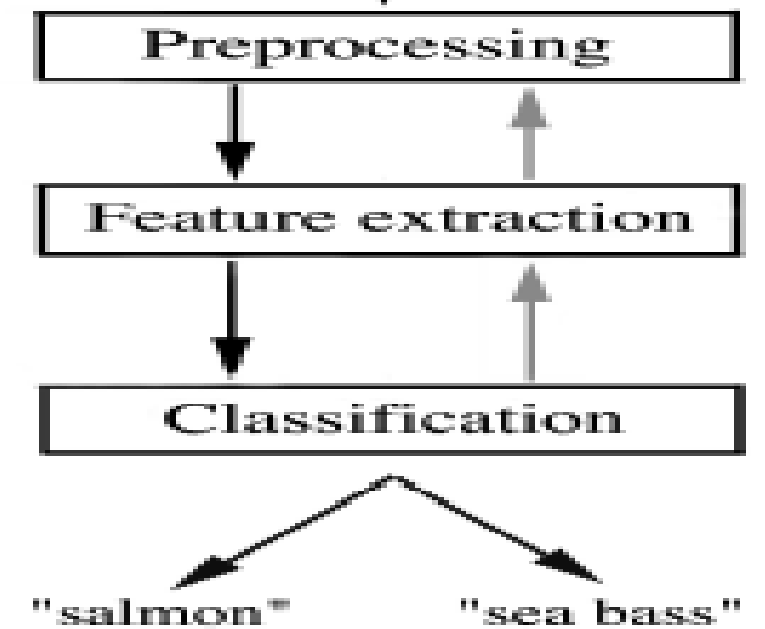
$$P(x \in \text{region}) = \int_{\text{region}} p(x) dx$$

Quelle dimensionnalité?
exemple...

Exemple de classification

- Séparer deux types de poissons (saumon et bar) sur un tapis roulant
 - **entrée** des données (caméra)
 - traitement d'image
 - **extraction des caractéristiques**/traits (largeur, longueur, luminosité, etc.)
 - design d'une **fonction de classification**:

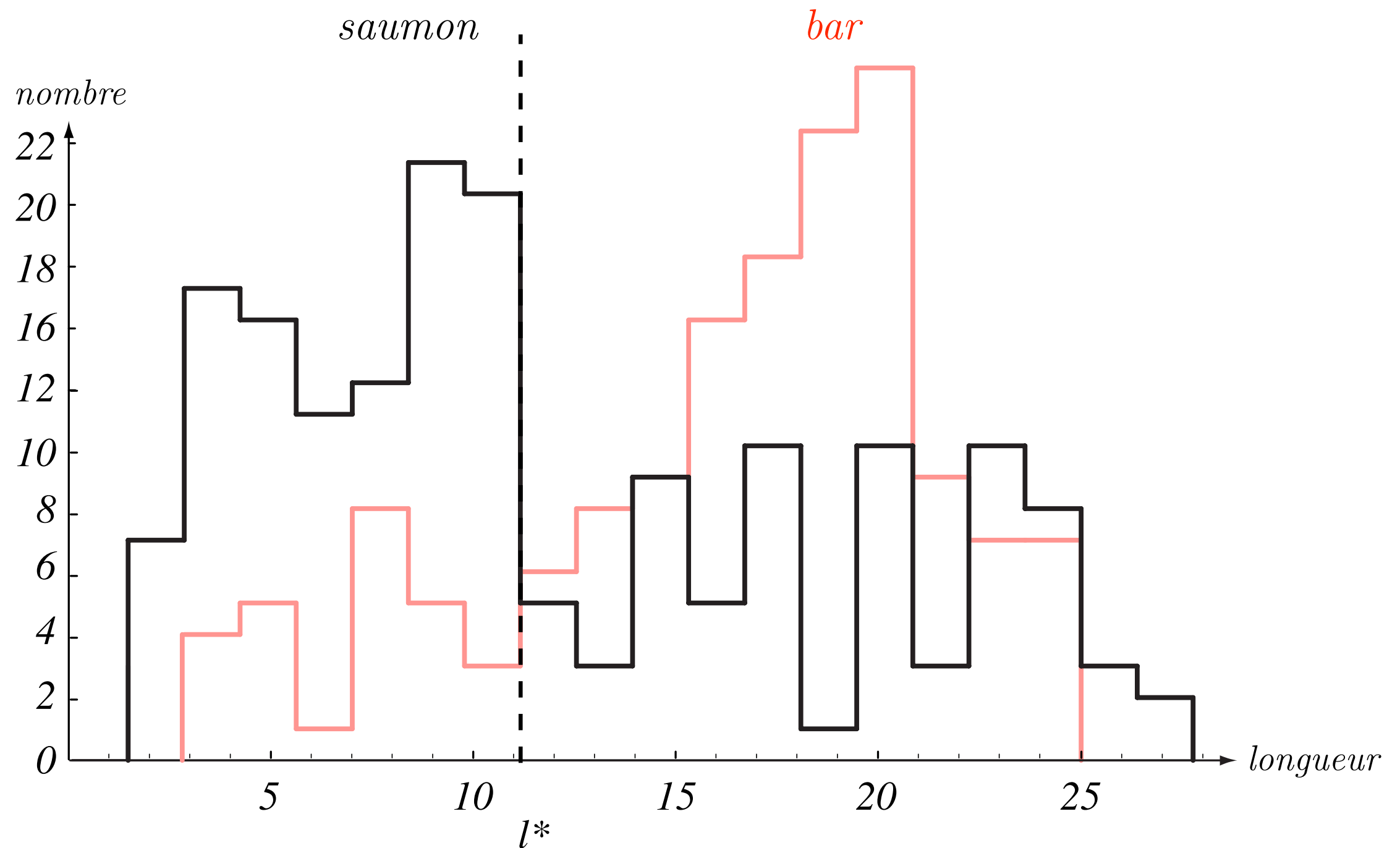
$$f : \{\text{vecteur des traits}\} \mapsto \{\text{saumon, bar}\}$$



Ex. de traits caractéristiques résultant du prétraitement
(preprocessing + feature extraction):

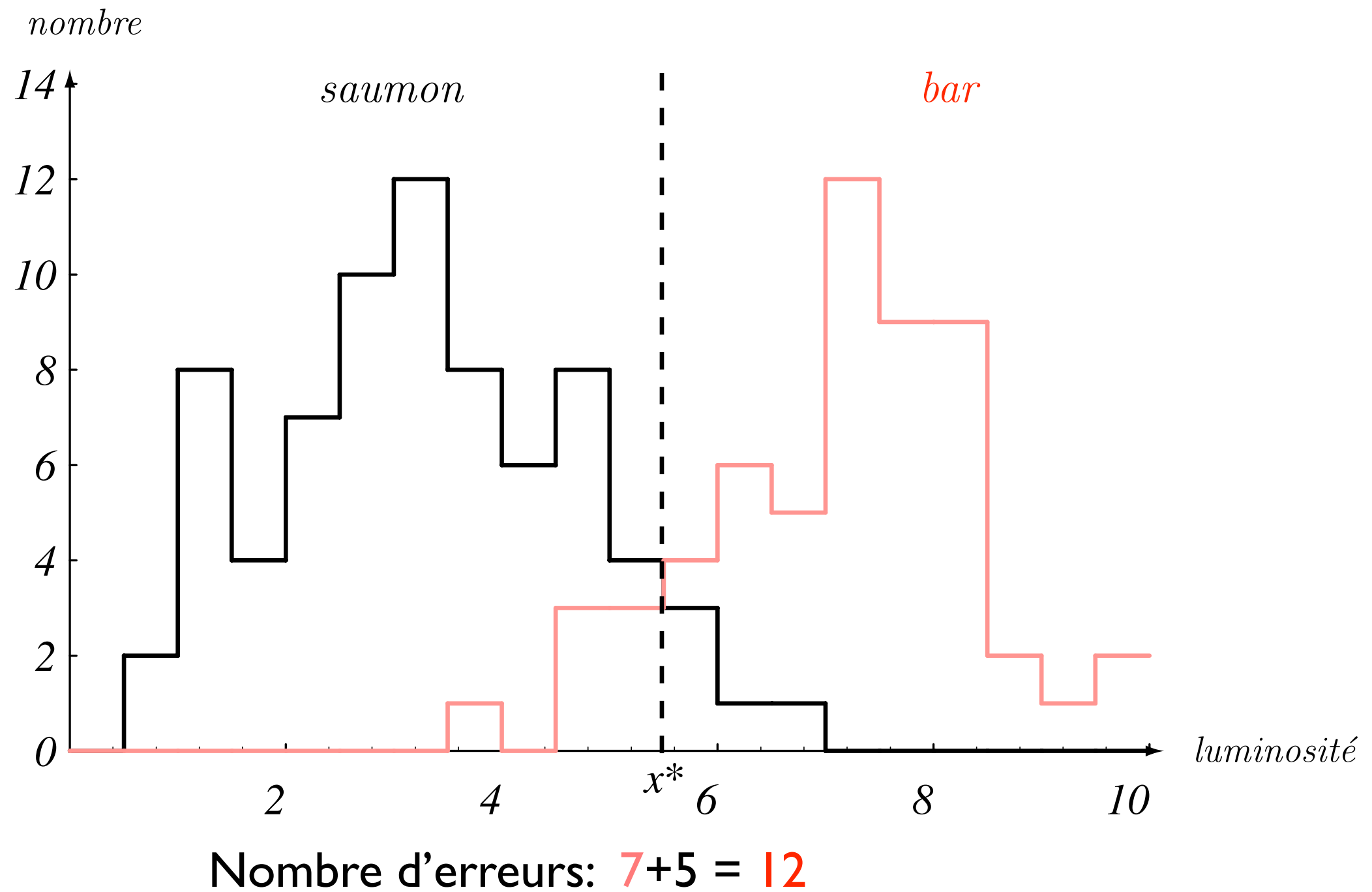
$x = (\text{longueur, largeur, luminosité, taille de la nageoire dorsale, position de la bouche, etc...})$

- Histogramme obtenu de l'ensemble d'entraînement
- erreur d'entraînement



Nombre d'erreurs: $26 + 69 = 95$ pour un classifieur linéaire

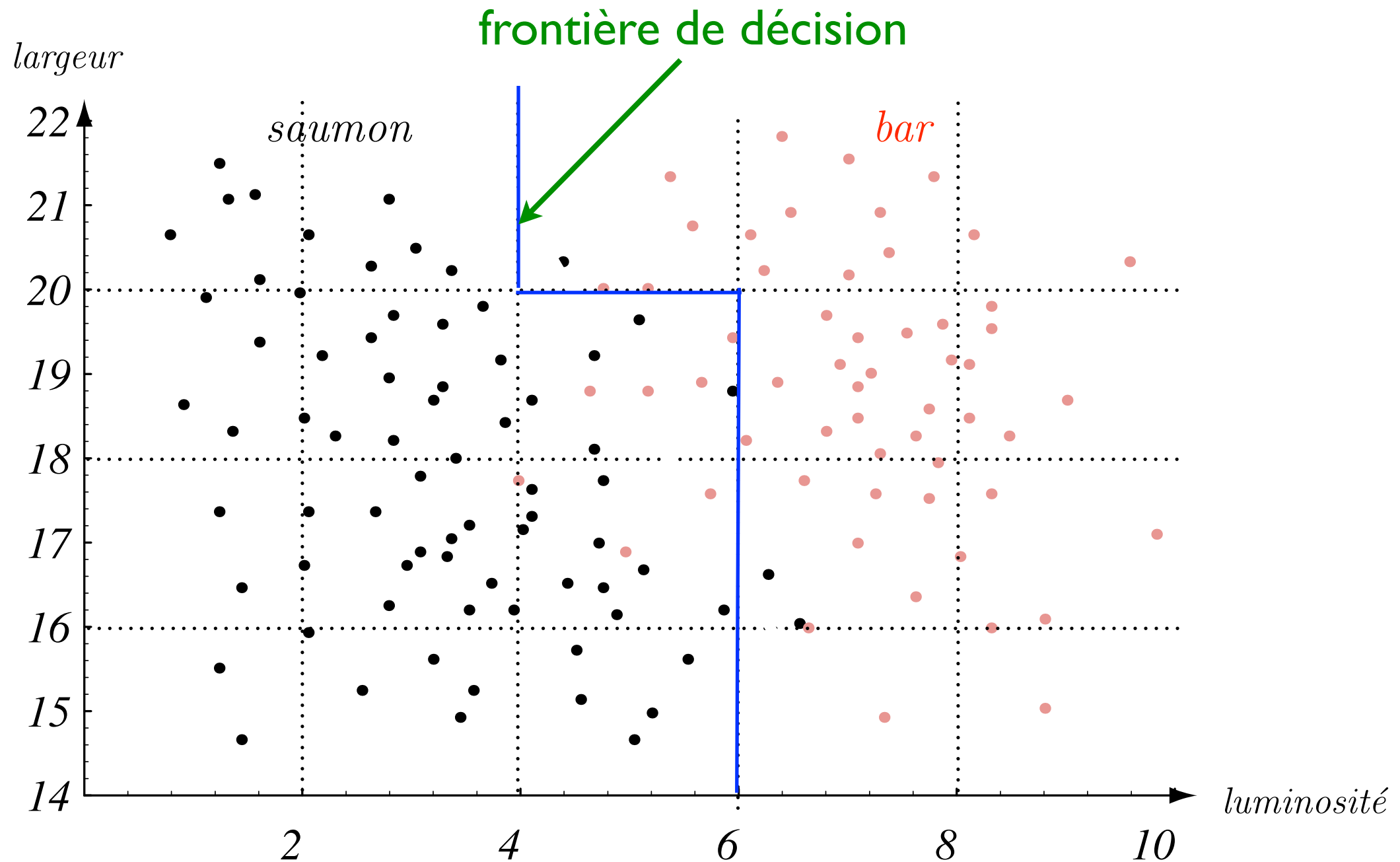
- Une autre variable/trait
 - coût de la mauvaise classification



- Deux variables

- vecteurs de traits, espace de traits, frontière de décision

pour un histogramme 2D

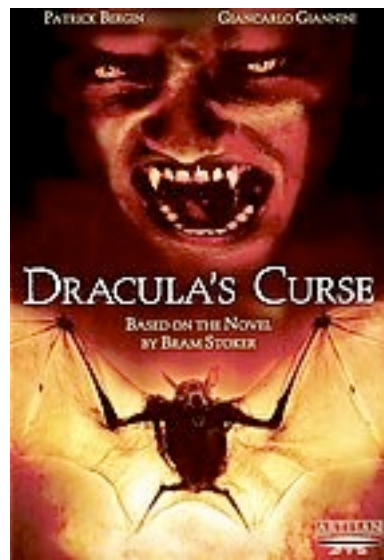


Nombre d'erreurs: = 10

- ◆ Plus de dimensions (traits caractéristiques)
c'est (généralement) plus d'information
pour prendre la bonne décision.
- ◆ Les classes en sont plus facilement
séparables
- ◆ C'est bien mais....

Malédiction (fléau) de la dimensionalité

CURSE OF DIMENSIONALITY



Ex: combien de cases pour un quadrillage découpé en 10 en dimension d ?

Si on a $n=100\,000$ points d'entraînement répartis \pm uniformément

- ◆ $d=1$: 10 cases
- ◆ $d=2$: $10 \times 10 = 100$ cases
- ◆ $d=3$: $10 \times 10 \times 10 = 1000$ cases
- ◆ $d=10$: $10^{10} = 10\,000\,000\,000$ cases
dix milliards!
- ◆ Pour un quadrillage où chaque dimension est découpée en m , on a m^d cases.

- ④ $d=1$: $100\,000/10 = 10\,000$ points/case
- ④ $d=2$: $100\,000/100 = 1\,000$ points/case
- ④ $d=3$: $100\,000/1\,000 = 100$ points/case
- ④ $d=10$: $100\,000/10^{10} = 10^{-5}$ points/case

- ④ $d=100$: $100\,000/10^{100} = 10^{-95}$ points/case

- ④ En haute dimension, la plupart des cases (où risque d'apparaître un point de test...) vont être **vide!!!**

La "taille" de l'espace explorable à modéliser croît exponentiellement avec la dimensionalité !

Sensibilité à la malédiction

- ◆ Les méthodes de type **histogramme** (quadrillage) fonctionnent **bien en faible dimension** (1, 2, voire 3)
- ◆ Mais sont **catastrophiques en haute dimension!**
- ◆ La **malédiction** de la dimensionalité affecte \pm tous les algorithmes d'apprentissage, mais **certains y sont beaucoup plus sensible** que d'autres.

Etapes de conception d'un algorithme d'apprentissage

- Compréhension intuitive de l'algorithme. Savoir l'expliquer en français!
- Formalisation mathématique de l'algorithme.
- Ecriture de l'algo sous forme de pseudo-code.
- Implémentation dans un langage/environnement de programmation.
- Entraînement/test de l'algo sur des problèmes simples en faible dimension, où on peut vérifier graphiquement si ça fait bien ce qu'on veut.
- Evaluation de performance sur des problèmes réels, et comparaison avec d'autres algorithmes concurrents.

Ex: Histogramme pour la classification

hyper-paramètres

Classifieur Histogramme2D ($m1, x1min, x1max, m2, x2min, x2max, nclasses$)

$C \leftarrow$ tenseur($m1, m2, nclasses$) initialisé à 0. paramètres

$taille1 = (x1max - x1min) / m1, \quad taille2 = (x2max - x2min) / m2$

ensemble d'entraînement

apprends(D_n): # "train"

Pour $(x, y) \in D_n$:

$i = \text{floor}((x[0] - x1min) / taille1), \quad j = \text{floor}((x[1] - x2min) / taille2)$

$C[i, j, y]++$

point de test

class_prob(x): # retourne un vecteur de probabilité que x soit de chaque classe.

$i = \text{floor}((x[0] - x1min) / taille1), \quad j = \text{floor}((x[1] - x2min) / taille2)$

comptes = $C[i, j]$

return comptes / sum(comptes)

point de test

f(x): # classe x . Retourne l'indice de la classe la plus probable

probs = class_prob(x)

return ArgMax[probs]

Complexité algorithmique

Histogramme pour la classification

Variables

n: nombre d'exemples,
d: dimensionnalité de l'entrée
C: nombre de classes
m: nombre de subdivisions,

Complexité mémoire (à l'entraînement et utilisation)

$$O(C m^d)$$

Complexité temps de calcul pour l'entraînement

$$O(d n)$$

Complexité temps de calcul à l'utilisation (f)

$$O(d+C)$$