CentraleSupélec

Statistique et apprentissage

Chargés de cours (ordre alphabétique) :

Julien Bect, Ziad Kobeissi, Gilles Faÿ, Laurent Le Brusquet,
Vincent Lescarret, Arshak Minasyan, Arthur Tenenhaus[†] & Xujia Zhu

[†] Coordinateur du cours

Cours 7/9

Classification : régression logistique. Quelques modèles pour l'apprentissage supervisé

Objectifs du cours 7

- Réaliser une classification par régression logistique
- Définir des mesures de performances pour la classification
- Savoir prédire à l'aide d'arbres de décision
- Savoir prédire à l'aide de réseaux de neurones

Plan du cours

Plan du cours

Plan du cours

Cadre mathématique et objectifs

Notations

- $(X_1, Y_1), \ldots, (X_n, Y_n) \overset{\text{iid}}{\sim} \mathrm{P}^{X,Y}$
- $\mathrm{P}^{X,Y}$: loi inconnue sur $\mathcal{X} \times \mathcal{Y}$
- $\mathcal{X} \subset \mathbb{R}^p, \quad \mathcal{Y} = \{0, 1, \ldots, K-1\}$
- sauf mention explicite : $K = 2$ (classification binaire)

Objectif

Construire une (bonne) fonction de prédiction $h : x \mapsto \{0, 1\}$.

Synonymes : fonction de classification, ou « classifieur ».

Objectifs de cette partie

- introduire la methode de la régression logistique
- définir des mesures de risque appropriées en classification

Cadre mathématique et objectifs

Notations

- $(X_1, Y_1), \ldots, (X_n, Y_n) \overset{\text{iid}}{\sim} \mathrm{P}^{X,Y}$
- $\mathrm{P}^{X,Y}$: loi inconnue sur $\mathcal{X} \times \mathcal{Y}$
- $\mathcal{X} \subset \mathbb{R}^p, \quad \mathcal{Y} = \{0, 1, \ldots, K-1\}$
- sauf mention explicite : $K = 2$ (classification binaire)

Objectif

Construire une (bonne) fonction de prédiction $h : x \mapsto \{0, 1\}$.

Synonymes : fonction de classification, ou « classifieur ».

Objectifs de cette partie

- introduire la methode de la régression logistique
- définir des mesures de risque appropriées en classification

Cadre mathématique et objectifs

Notations

- $(X_1, Y_1), \ldots, (X_n, Y_n) \overset{\text{iid}}{\sim} \mathrm{P}^{X,Y}$
- $\mathrm{P}^{X,Y}$: loi inconnue sur $\mathcal{X} \times \mathcal{Y}$
- $\mathcal{X} \subset \mathbb{R}^p, \quad \mathcal{Y} = \{0, 1, \ldots, K-1\}$
- sauf mention explicite : $K = 2$ (classification binaire)

Objectif

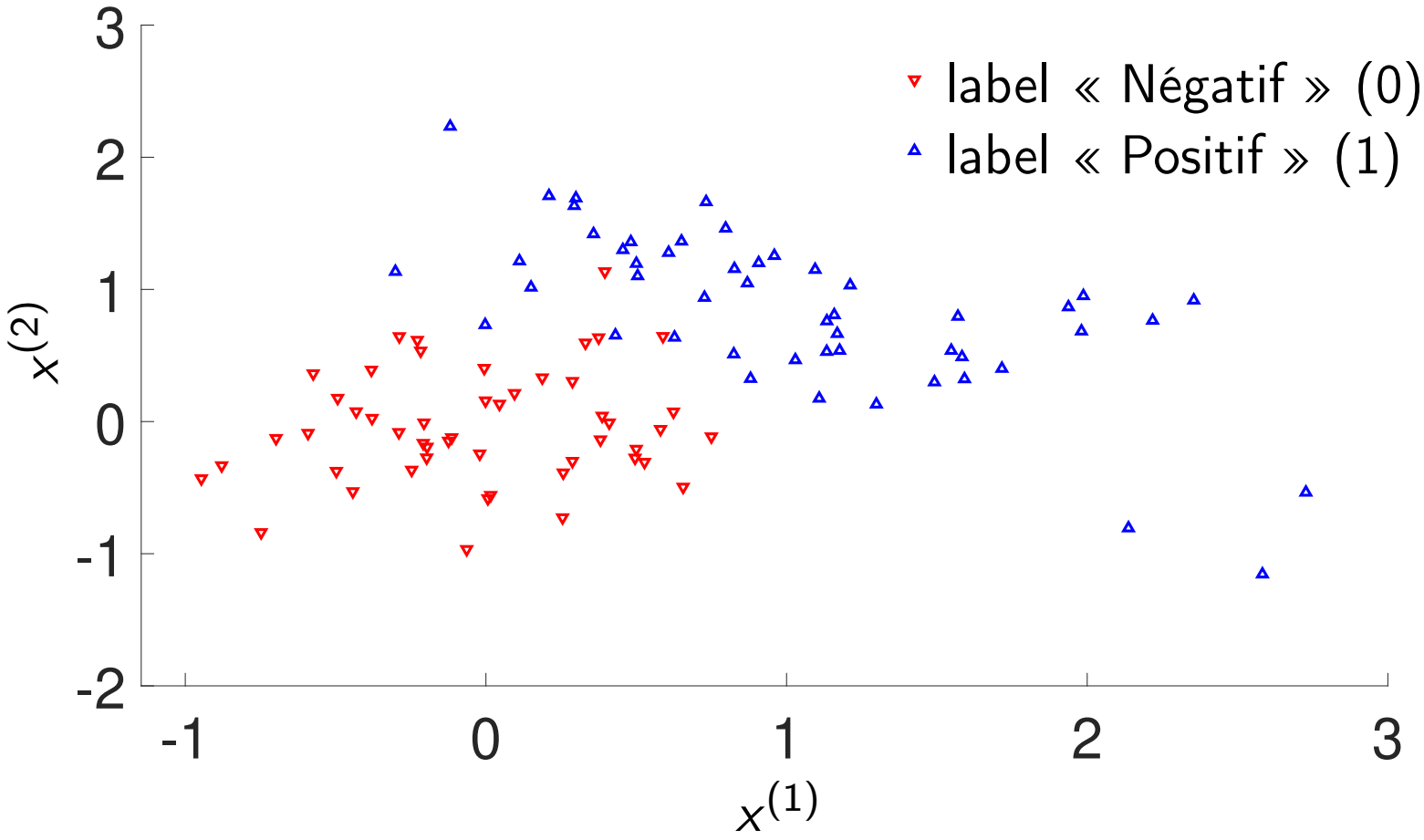
Construire une (bonne) fonction de prédiction $h : x \mapsto \{0, 1\}$.

Synonymes : fonction de classification, ou « classifieur ».

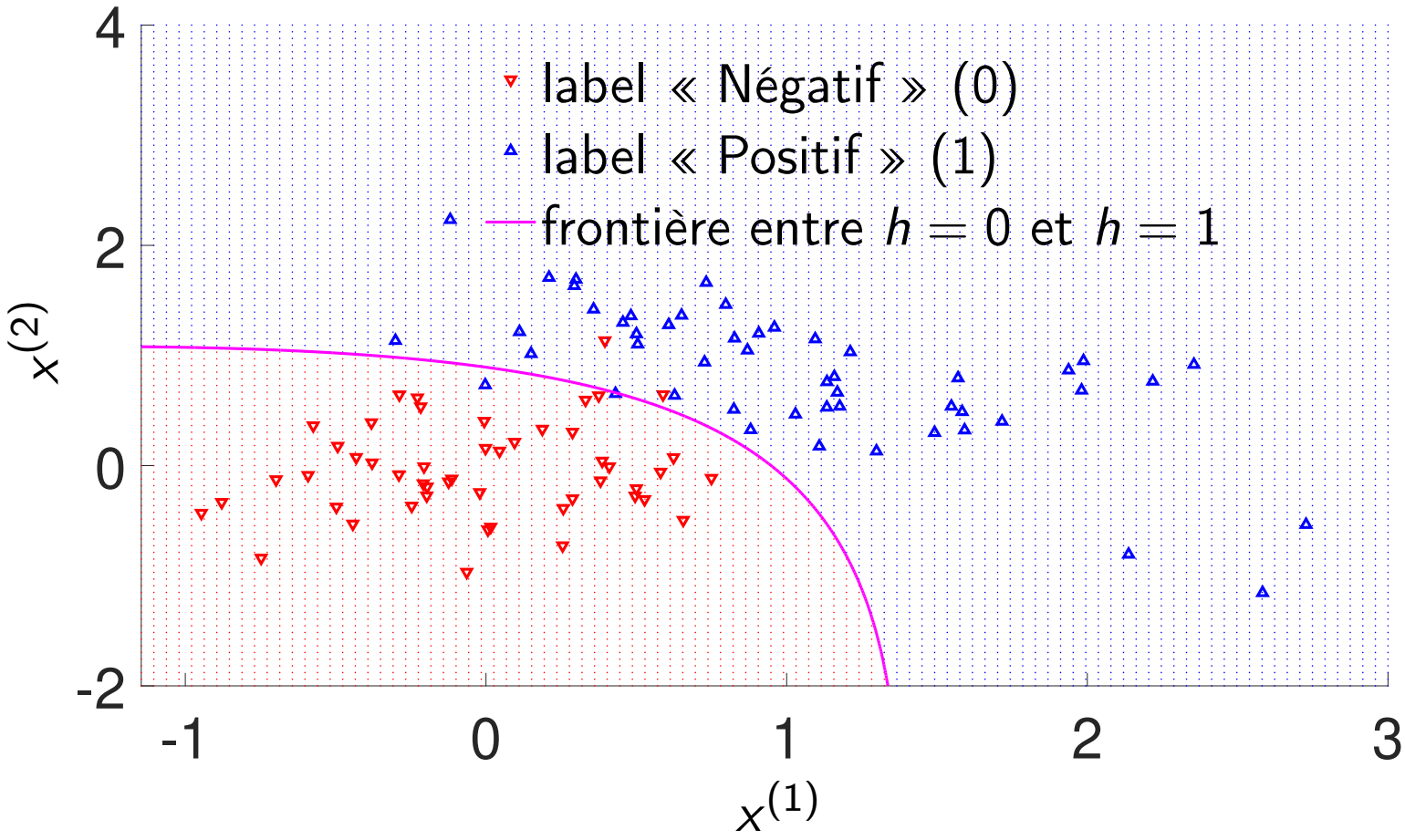
Objectifs de cette partie

- introduire la methode de la régression logistique
- définir des mesures de risque appropriées en classification

Exemple avec deux variables explicatives ($p = 2$)



Avant-goût de la suite : un classifieur possible



Plan du cours

Rappel : fonction de perte et risque

Définition : risque (erreur de généralisation)

Etant donnée une fonction de perte $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ et une fonction de prédiction $h : \mathcal{X} \to \mathcal{Y}$, on définit le risque, ou erreur de généralisation :

$$R(h) = \mathbb{E}\left(L(Y, h(X))\right),$$

l'espérance portant sur le couple (X, Y).

⚠ Ce risque dépend de la loi inconnue :

$$R(h) = \iint_{\mathcal{X}\times\mathcal{Y}} L(y, h(x)) \, \mathrm{P}^{X,Y}(\mathrm{d}x, \mathrm{d}y).$$

Rappel : fonction de perte et risque

Définition : risque (erreur de généralisation)

Etant donnée une fonction de perte $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ et une fonction de prédiction $h : \mathcal{X} \to \mathcal{Y}$, on définit le risque, ou erreur de généralisation :

$$R(h) = \mathbb{E}\left(L(Y, h(X))\right),$$

l'espérance portant sur le couple (X, Y).

⚠ Ce risque dépend de la loi inconnue :

$$R(h) = \iint_{\mathcal{X}\times\mathcal{Y}} L(y, h(x)) \, \mathrm{P}^{X,Y}(\mathrm{d}x, \mathrm{d}y).$$

Fonctions de classification dures et douces

Considérons un problème à deux classes : $\mathcal{Y} = \{0, 1\}$.

Définition : fonction de classification dure

On appelle fonction de classification dure une fonction mesurable $h : \mathcal{X} \to \mathcal{Y}$ de l'espace des exemples dans l'espace des étiquettes.

Remarque : cas particulier de fonction de prédiction, cf. cours 6.

Définition : fonction de classification douce

On appelle fonction de classification douce une fonction mesurable $h : \mathcal{X} \to [0, 1]$ de l'espace des exemples dans $[0, 1]$.

Partant d'une fonction de classification douce $h : \mathcal{X} \to [0, 1]$, on peut construire une famille de fonctions de classification dures, de la forme $x \mapsto \mathbb{1}_{h(x) \geq \delta}$, pour $\delta \in [0, 1]$.

Fonctions de classification dures et douces

Considérons un problème à deux classes : $\mathcal{Y} = \{0, 1\}$.

Définition : fonction de classification dure

On appelle fonction de classification dure une fonction mesurable $h : \mathcal{X} \to \mathcal{Y}$ de l'espace des exemples dans l'espace des étiquettes.

Remarque : cas particulier de fonction de prédiction, cf. cours 6.

Définition : fonction de classification douce

On appelle fonction de classification douce une fonction mesurable $h : \mathcal{X} \to [0, 1]$ de l'espace des exemples dans $[0, 1]$.

Partant d'une fonction de classification douce $h : \mathcal{X} \to [0, 1]$, on peut construire une famille de fonctions de classification dures, de la forme $x \mapsto \mathbb{1}_{h(x) \geq \delta}$, pour $\delta \in [0, 1]$.

Fonctions de classification dures et douces

Considérons un problème à deux classes : $\mathcal{Y} = \{0, 1\}$.

Définition : fonction de classification dure

On appelle fonction de classification dure une fonction mesurable $h : \mathcal{X} \to \mathcal{Y}$ de l'espace des exemples dans l'espace des étiquettes.

Remarque : cas particulier de fonction de prédiction, cf. cours 6.

Définition : fonction de classification douce

On appelle fonction de classification douce une fonction mesurable $h : \mathcal{X} \to [0, 1]$ de l'espace des exemples dans $[0, 1]$.

Partant d'une fonction de classification douce $h : \mathcal{X} \to [0, 1]$, on peut construire une famille de fonctions de classification dures, de la forme $x \mapsto \mathbb{1}_{h(x) \geq \delta}$, pour $\delta \in [0, 1]$.

Fonctions de perte usuelles

Définition : Perte 0/1 pour la classification dure

$$\begin{aligned} L : \mathcal{Y} \times \mathcal{Y} &\to \mathbb{R}_+ \\ (y, \tilde{y}) &\mapsto L(y, \tilde{y}) = \mathbb{1}_{y \neq \tilde{y}}. \end{aligned}$$

- $R(h) = \mathbb{P}(Y \neq h(X))$ est la probabilité de mauvais classement.

Définition : Perte logarithmique pour la classification douce

$$\begin{aligned} L : \mathcal{Y} \times [0,1] &\to \mathbb{R}_+ \\ (y, \tilde{y}) &\mapsto L(y, \tilde{y}) = \begin{cases} -\ln(\tilde{y}) & \text{si } y = 1, \\ -\ln(1 - \tilde{y}) & \text{si } y = 0. \end{cases} \end{aligned}$$

Fonctions de perte usuelles

Définition : Perte 0/1 pour la classification dure

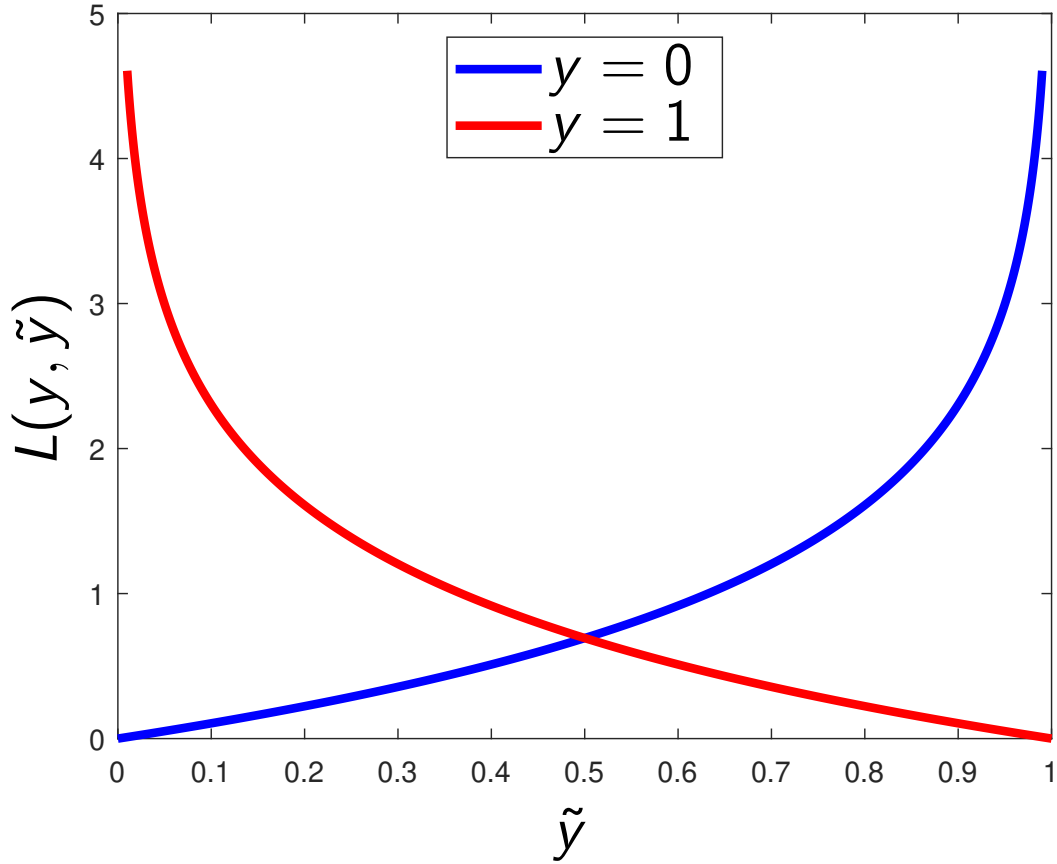
$$\begin{aligned} L : \mathcal{Y} \times \mathcal{Y} &\to \mathbb{R}_+ \\ (y, \tilde{y}) &\mapsto L(y, \tilde{y}) = \mathbb{1}_{y \neq \tilde{y}}. \end{aligned}$$

- $R(h) = \mathbb{P}(Y \neq h(X))$ est la probabilité de mauvais classement.

Définition : Perte logarithmique pour la classification douce

$$\begin{aligned} L : \mathcal{Y} \times [0,1] &\to \mathbb{R}_+ \\ (y, \tilde{y}) &\mapsto L(y, \tilde{y}) = \begin{cases} -\ln(\tilde{y}) & \text{si } y = 1, \\ -\ln(1-\tilde{y}) & \text{si } y = 0. \end{cases} \end{aligned}$$

Fonctions de perte usuelles (suite)



Remarque : pour les deux fonctions de pertes,

- $L(y, \tilde{y}) \geq 0$,
- $L(y, \tilde{y}) = 0 \Leftrightarrow \tilde{y} = y$.

Fonctions de classifications optimales

Proposition

$h : \mathcal{X} \to \mathcal{Y}$ est optimale pour la perte 0/1 ssi, P^X-pp,

- $h(x) = 1$ lorsque $\mathrm{P}(Y = 1 \mid X = x) > \frac{1}{2}$,
- $h(x) = 0$ lorsque $\mathrm{P}(Y = 1 \mid X = x) < \frac{1}{2}$.

où l'on a introduit la notation $\mathrm{P}(A \mid X = x) = \mathbb{E}(\mathbb{1}_A \mid X = x)$.

démonstration : voir TD 7

Par exemple, $x \mapsto \mathbb{1}_{\mathrm{P}(Y=1|X=x) \geq \frac{1}{2}}$ est optimale.

Remarque : une formule plus générale peut être établie pour une perte asymétrique ($L(0,1) \neq L(1,0)$). Voir polycopié.

Fonctions de classifications optimales (suite)

Proposition

$h : \mathcal{X} \to [0,1]$ est optimale pour la perte logarithmique ssi, P^X-pp,

$$h(x) \;=\; \mathrm{P}\left(Y = 1 \mid X = x\right).$$

démonstration : exercice 1

Remarque : puisque Y est à valeurs dans $\{0,1\}$, on a :

$$\mathrm{P}\left(Y = 1 \mid X = x\right) \;=\; \mathbb{E}\left(Y \mid X = x\right).$$

☞ classification douce + perte logarithmique $\approx$ régression.

(D'où le nom de la prochaine méthode que nous allons étudier !)

Fonctions de classifications optimales (suite)

Proposition

$h : \mathcal{X} \to [0,1]$ est optimale pour la perte logarithmique ssi, P^X-pp,

$$h(x) \;=\; \mathrm{P}\left(Y = 1 \mid X = x\right).$$

démonstration : exercice 1

Remarque : puisque Y est à valeurs dans $\{0,1\}$, on a :

$$\mathrm{P}\left(Y = 1 \mid X = x\right) \;=\; \mathbb{E}\left(Y \mid X = x\right).$$

☞ classification douce + perte logarithmique $\approx$ régression.

(D'où le nom de la prochaine méthode que nous allons étudier !)

Plan du cours

Matrice de confusion & grandeurs dérivées

	Réalité Négatif (N)	Réalité Positif (P)
Prédiction Négatif	Vrais Négatifs (VN)	Faux Négatifs (FN)
Prédiction Positif	Faux Positifs (FP)	Vrais Positifs (VP)

Taux de Vrais Positifs

$$TVP = \frac{VP}{P} = \frac{VP}{VP + FN}$$

(également appelé sensibilité)

Taux de Vrais Négatifs

$$TVN = \frac{VN}{N} = \frac{VN}{VN + FP}$$

(également appelé spécificité)

Matrice de confusion & grandeurs dérivées

	Réalité Négatif (N)	Réalité Positif (P)
Prédiction Négatif	Vrais Négatifs (VN)	Faux Négatifs (FN)
Prédiction Positif	Faux Positifs (FP)	Vrais Positifs (VP)

Taux de Vrais Positifs

$$TVP = \frac{VP}{P} = \frac{VP}{VP + FN}$$

(également appelé sensibilité)

Taux de Vrais Négatifs

$$TVN = \frac{VN}{N} = \frac{VN}{VN + FP}$$

(également appelé spécificité)

Matrice de confusion & grandeurs dérivées (suite)

Autre terminologie, issue du traitement du signal :

- $1 - TVP$ est le taux de non-détections (taux de faux négatifs)
- $1 - TVN$ est le taux de fausses alarmes (taux de faux positifs)

Application aux fonctions de classification douces

- Rappel : à toute fonction de classification douce h on peut associer une famille de fonctions de classification dures

$$h_\delta : x \mapsto \mathbb{1}_{h(x) \geq \delta}, \qquad \delta \in [0, 1] .$$

- La valeur de δ influe sur le compromis TVN/TVP
  - quand $\delta \nearrow$, $TVN \nearrow$, et $TVP \searrow$

Matrice de confusion & grandeurs dérivées (suite)

Autre terminologie, issue du traitement du signal :

- $1 - TVP$ est le taux de non-détections (taux de faux négatifs)
- $1 - TVN$ est le taux de fausses alarmes (taux de faux positifs)

Application aux fonctions de classification douces

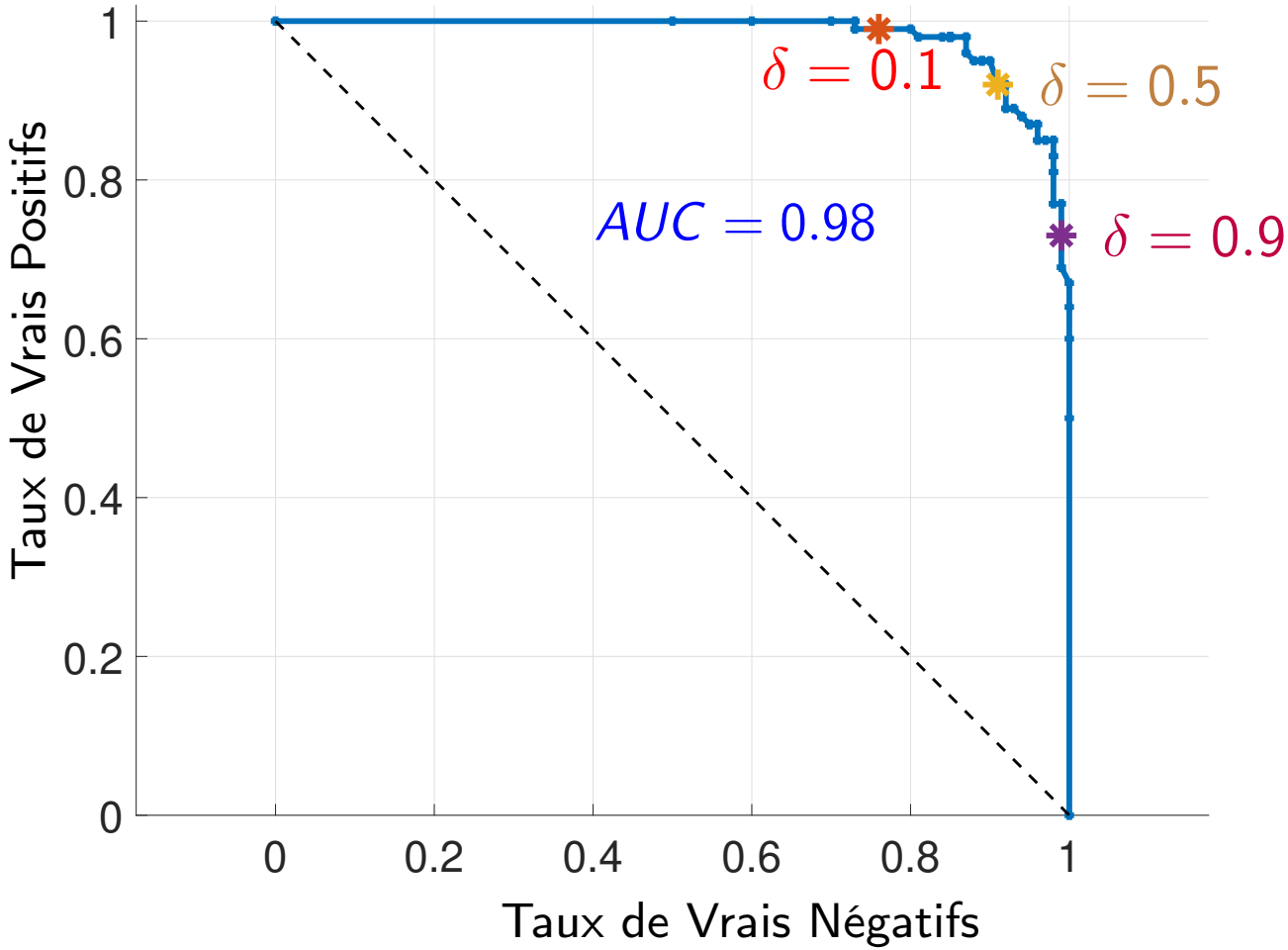
- Rappel : à toute fonction de classification douce h on peut associer une famille de fonctions de classification dures

$$h_\delta : x \mapsto \mathbb{1}_{h(x) \geq \delta}, \qquad \delta \in [0, 1] .$$

- La valeur de δ influe sur le compromis TVN/TVP
  - quand $\delta \nearrow$, $TVN \nearrow$, et $TVP \searrow$

Courbe ROC (Receiver Operating Characteristic)

- un outil d'aide à la décision (choix de δ)
- un outil de comparaison de classifieurs
- grandeur dérivée : *AUC* = Area Under the Curve



Plan du cours

Plan du cours

Un modèle linéaire pour la classification douce

On considère un problème de classification (binaire)

- $\mathcal{X} \subset \mathbb{R}^p$, $\mathcal{Y} = \{0, 1\}$.

On choisit la perte logarithmique :

- l'objectif est d'approcher la fonction de classification optimale

$$h^*(x) \;=\; \mathbb{P}(Y = 1 \mid X = x)\,.$$

La régression logistique consiste à utiliser des fonctions de classification douces de la forme

$$h(x) \;=\; s\left(\beta_0 + \beta^\top x\right),$$

avec $\beta_0 \in \mathbb{R}$, $\beta \in \mathbb{R}^p$, et $s(t) = e^t/(1 + e^t)$ la fonction logistique.

Un modèle linéaire pour la classification douce

On considère un problème de classification (binaire)

- $\mathcal{X} \subset \mathbb{R}^p$, $\mathcal{Y} = \{0, 1\}$.

On choisit la perte logarithmique :

- l'objectif est d'approcher la fonction de classification optimale

$$h^*(x) \;=\; \mathbb{P}(Y = 1 \mid X = x)\,.$$

La régression logistique consiste à utiliser des fonctions de classification douces de la forme

$$h(x) \;=\; s\left(\beta_0 + \beta^\top x\right),$$

avec $\beta_0 \in \mathbb{R}$, $\beta \in \mathbb{R}^p$, et $s(t) = e^t/(1 + e^t)$ la fonction logistique.

Un modèle linéaire pour la classification douce

On considère un problème de classification (binaire)

- $\mathcal{X} \subset \mathbb{R}^p$, $\mathcal{Y} = \{0, 1\}$.

On choisit la perte logarithmique :

- l'objectif est d'approcher la fonction de classification optimale

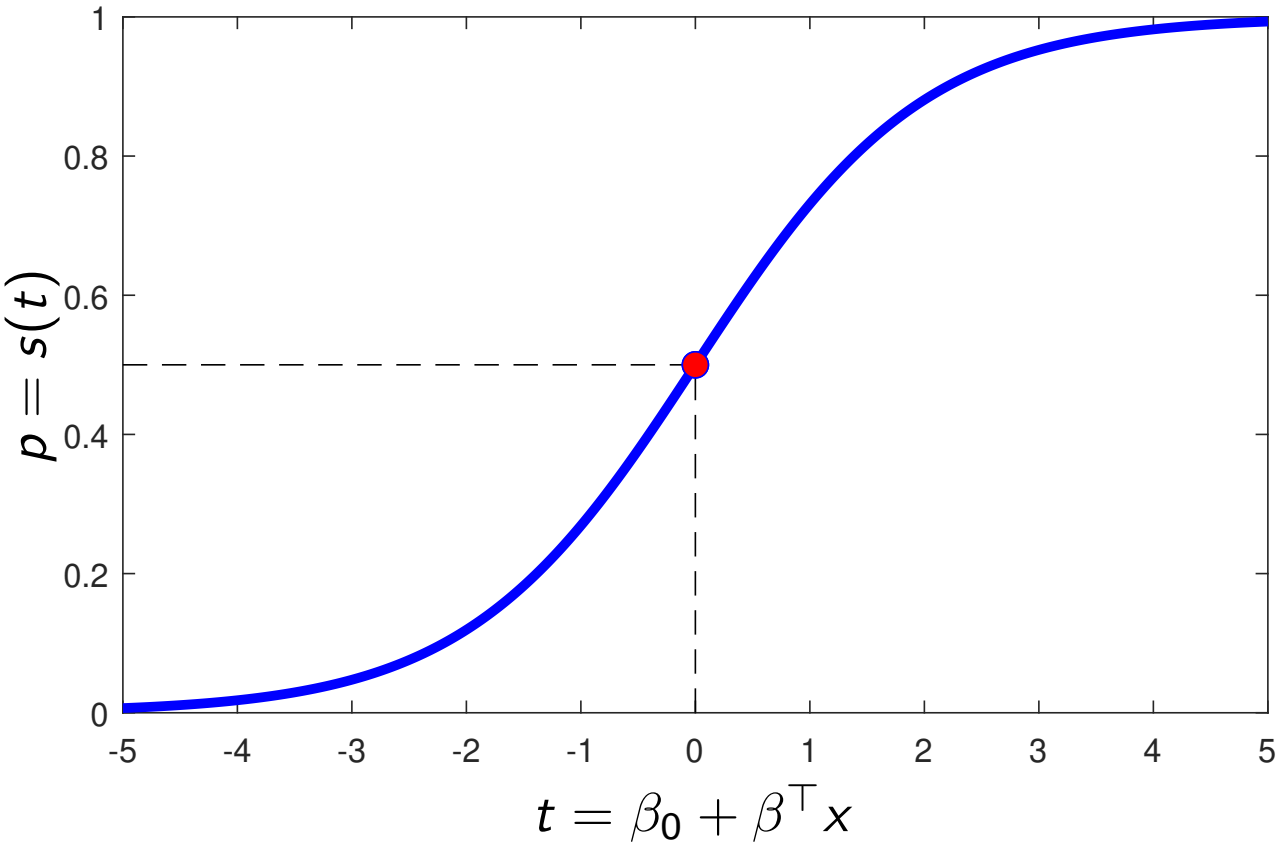
$$h^*(x) \;=\; \mathbb{P}(Y = 1 \mid X = x)\,.$$

La régression logistique consiste à utiliser des fonctions de classification douces de la forme

$$h(x) \;=\; s\left(\beta_0 + \beta^\top x\right),$$

avec $\beta_0 \in \mathbb{R}$, $\beta \in \mathbb{R}^p$, et $s(t) = e^t/(1 + e^t)$ la fonction logistique.

La fonction logistique



➠ définit une correspondance : $\beta_0 + \beta^\top x \in \mathbb{R} \longleftrightarrow$ proba $p \in]0, 1[$

Aussi connue sous le nom de fonction sigmoïde.

Un modèle linéaire pour la classification douce (suite)

De manière équivalente,

$$\text{logit}\left(h(x)\right) \;=\; \beta_0 + \beta^\top x$$

avec

$$\begin{array}{rccc} \text{logit}: &]0,1[& \to & \mathbb{R} \\ & p & \mapsto & \ln\left(\dfrac{p}{1-p}\right) \end{array}$$

la "fonction logit".

Propriétés

- La fonction logistique s est une bijection strictement croissante et C^∞ de $\mathbb{R}$ dans $]0,1[$.
- La fonction logit est la bijection reciproque : elle est strictement croissante et C^∞ de $]0,1[$ dans $\mathbb{R}$.

Un modèle linéaire pour la classification douce (suite)

De manière équivalente,

$$\text{logit}\left(h(x)\right) \;=\; \beta_0 + \beta^\top x$$

avec

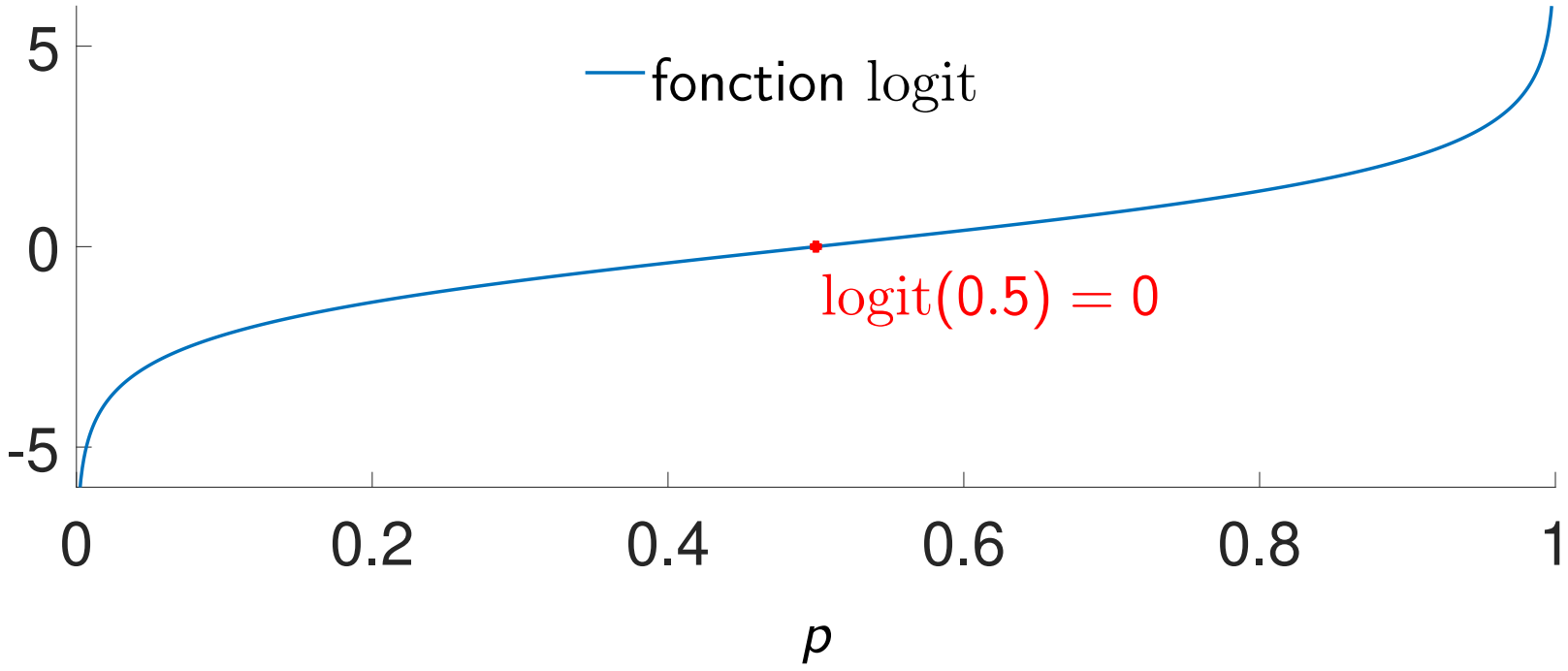
$$\begin{array}{rccc} \text{logit}: &]0,1[& \to & \mathbb{R} \\ & p & \mapsto & \ln\left(\dfrac{p}{1-p}\right) \end{array}$$

la "fonction logit".

Propriétés

- La fonction logistique s est une bijection strictement croissante et C^∞ de $\mathbb{R}$ dans $]0,1[$.
- La fonction logit est la bijection reciproque : elle est strictement croissante et C^∞ de $]0,1[$ dans $\mathbb{R}$.

La fonction logit



➠ définit une correspondance : proba $p \in]0,1[\longleftrightarrow \beta_0 + \beta^\top x \in \mathbb{R}$

De la classification douce à la classification dure

Etant donné une fonction de classification douce de la forme

$$h(x) \;=\; s\left(\beta_0 + \beta^\top x\right),$$

et un seuil de décision $\delta \in [0,1]$, on pose :

$$h_\delta(x) \;=\; \mathbb{1}_{h(x) \geq \delta}.$$

☞ h_δ sépare les classes dans $\mathcal{X}$ par un hyperplan affine :

$$h_\delta(x) = 1 \quad \Longleftrightarrow \quad \beta_0 + \beta^\top x \geq \mathrm{logit}(\delta)$$

Pour la perte 0-1, la valeur $\delta = \frac{1}{2}$ est généralement utilisée.

De la classification douce à la classification dure

Etant donné une fonction de classification douce de la forme

$$h(x) \;=\; s\left(\beta_0 + \beta^\top x\right),$$

et un seuil de décision $\delta \in [0, 1]$, on pose :

$$h_\delta(x) \;=\; \mathbb{1}_{h(x) \geq \delta}.$$

☞ h_δ sépare les classes dans $\mathcal{X}$ par un hyperplan affine :

$$h_\delta(x) = 1 \quad \Longleftrightarrow \quad \beta_0 + \beta^\top x \geq \mathrm{logit}(\delta)$$

Pour la perte 0-1, la valeur $\delta = \frac{1}{2}$ est généralement utilisée.

Plan du cours

Minimisation du risque empirique

Allègement des notations : $x \to \begin{pmatrix} 1 \\ x \end{pmatrix}$ et $\beta \to \begin{pmatrix} \beta_0 \\ \beta \end{pmatrix}$

$$\Rightarrow \quad h(x) = \frac{\exp\left(\beta^\top x\right)}{1 + \exp\left(\beta^\top x\right)}$$

On choisit β par minimisation du risque empirique :

$$\hat{\beta} \;=\; \text{argmin}_{\beta \in \mathbb{R}^{p+1}} \frac{1}{n} \sum_{i=1}^{n} L\left(Y_i,\, h(X_i)\right),$$

où L désigne la perte logarithmique.

Minimisation du risque empirique

Allègement des notations : $x \to \begin{pmatrix} 1 \\ x \end{pmatrix}$ et $\beta \to \begin{pmatrix} \beta_0 \\ \beta \end{pmatrix}$

$$\Rightarrow \quad h(x) = \frac{\exp\left(\beta^\top x\right)}{1 + \exp\left(\beta^\top x\right)}$$

On choisit β par minimisation du risque empirique :

$$\hat{\beta} \;=\; \text{argmin}_{\beta \in \mathbb{R}^{p+1}} \frac{1}{n} \sum_{i=1}^{n} L\left(Y_i,\, h(X_i)\right),$$

où L désigne la perte logarithmique.

Minimisation du risque empirique (suite)

Équivalence entre minimisation du risque empirique et EMV

$$\sum_{i=1}^{n} L(y_i, h(x_i)) \;=\; -\ln\Big(\underbrace{\prod_{i=1}^{n} (h(x_i))^{y_i} (1 - h(x_i))^{1-y_i}}_{} \Big)$$

Interprétation : $\hat{\beta}$ est l'EMV dans le modèle paramétrique

$$Y_i | X_i \overset{\text{iid}}{\sim} \mathrm{Ber}(h(X_i)), \qquad h(x) = \frac{\exp\left(\beta^\top x\right)}{1 + \exp\left(\beta^\top x\right)}, \qquad \beta \in \mathbb{R}^{p+1}.$$

Minimisation du risque empirique (suite)

Équivalence entre minimisation du risque empirique et EMV

$$\sum_{i=1}^{n} L(y_i, h(x_i)) = -\ln\Big(\underbrace{\prod_{i=1}^{n} (h(x_i))^{y_i} (1 - h(x_i))^{1-y_i}}_{\text{vraisemblance } \mathcal{L}(\beta; \underline{X}, \underline{Y})} \Big)$$

Interprétation : $\hat{\beta}$ est l'EMV dans le modèle paramétrique

$$Y_i | X_i \overset{\text{iid}}{\sim} \mathrm{Ber}(h(X_i)), \qquad h(x) = \frac{\exp\left(\beta^\top x\right)}{1 + \exp\left(\beta^\top x\right)}, \qquad \beta \in \mathbb{R}^{p+1}.$$

Log-vraisemblance

Log-vraisemblance (voir TD)

$$\begin{aligned}\ell(\beta) &= \ln \mathcal{L}(\beta; \underline{X}, \underline{Y}) \\ &= \sum_{i=1}^{n} \left\{ Y_i \, \beta^\top X_i - \ln\left(1 + \exp(\beta^\top X_i)\right) \right\}\end{aligned}$$

Maximisation de ℓ

Elle s'effectue à l'aide d'un algorithme d'optimisation numérique

➠ par exemple, l'algorithme de Newton-Raphson

Log-vraisemblance

Log-vraisemblance (voir TD)

$$\begin{aligned}\ell(\beta) &= \ln \mathcal{L}(\beta; \underline{X}, \underline{Y}) \\ &= \sum_{i=1}^{n} \left\{ Y_i \, \beta^\top X_i - \ln\left(1 + \exp(\beta^\top X_i)\right) \right\}\end{aligned}$$

Maximisation de ℓ

Elle s'effectue à l'aide d'un algorithme d'optimisation numérique

➠ par exemple, l'algorithme de Newton-Raphson

Log-vraisemblance

Log-vraisemblance (voir TD)

$$\begin{aligned}\ell(\beta) &= \ln \mathcal{L}(\beta; \underline{X}, \underline{Y}) \\ &= \sum_{i=1}^{n} \left\{ Y_i \, \beta^\top X_i - \ln\left(1 + \exp(\beta^\top X_i)\right) \right\}\end{aligned}$$

Maximisation de ℓ

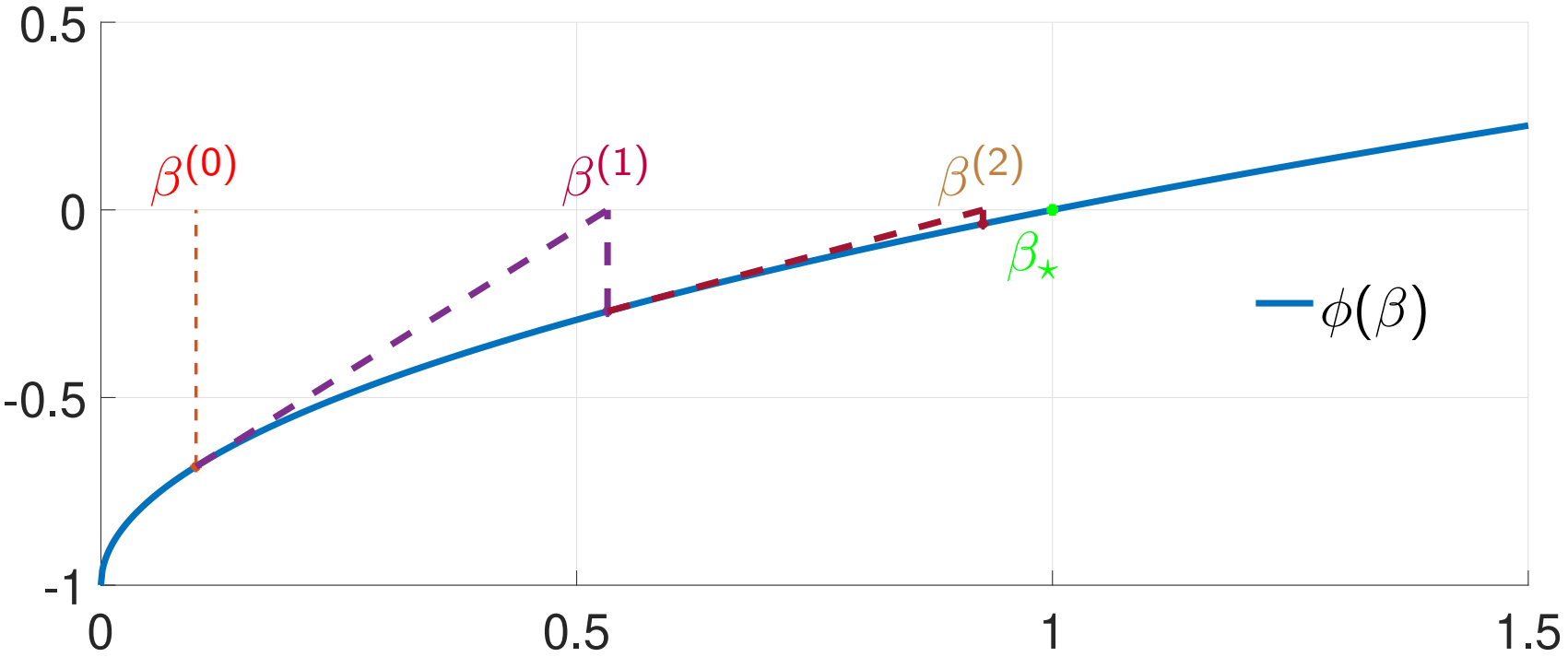
Elle s'effectue à l'aide d'un algorithme d'optimisation numérique

➠ par exemple, l'algorithme de Newton-Raphson

Rappel : Algorithme de Newton-Raphson unidimensionel

Soit $\phi : \mathbb{R} \to \mathbb{R}$. On cherche β tel que $\phi(\beta) = 0$
L'algorithme de Newton-Raphson est itératif :

- initialisation : $\beta^{(0)}$
- récurrence : $\beta^{(k+1)} = \beta^{(k)} - \dfrac{\phi\left(\beta^{(k)}\right)}{\phi'\left(\beta^{(k)}\right)}$



Maximisation de ℓ par la méthode de Newton-Raphson

Même algorithme mais cette fois en dimension $p+1$, avec :

- $\phi \to \nabla_\beta \ell$
- $\phi' \to \nabla^2_\beta \ell$

D'où la récurrence :

$$\beta^{(k+1)} = \beta^{(k)} - \left[\nabla^2_\beta \ell\left(\beta^{(k)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(k)}\right)$$

Sous les conditions suivantes :

- $\nabla^2_\beta \ell\,(.)$ est une application lipschitzienne,
- $\nabla^2_\beta \ell\left(\beta^{(0)}\right)$ est inversible,
- $h_0 = \left[\nabla^2_\beta \ell\left(\beta^{(0)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(0)}\right)$ est suffisamment[†] petit,

l'algorithme converge vers un point $\beta^\star$ tel que $\nabla_\beta \ell\,(\beta^\star) = 0$.

[†] voir la page Wikipédia sur le théorème de Kantorovich pour une quantification du « suffisamment »

Maximisation de ℓ par la méthode de Newton-Raphson

Même algorithme mais cette fois en dimension $p+1$, avec :

- $\phi \to \nabla_\beta \ell$
- $\phi' \to \nabla^2_\beta \ell$

D'où la récurrence :

$$\beta^{(k+1)} = \beta^{(k)} - \left[\nabla^2_\beta \ell\left(\beta^{(k)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(k)}\right)$$

Sous les conditions suivantes :

- $\nabla^2_\beta \ell\,(.)$ est une application lipschitzienne,
- $\nabla^2_\beta \ell\left(\beta^{(0)}\right)$ est inversible,
- $h_0 = \left[\nabla^2_\beta \ell\left(\beta^{(0)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(0)}\right)$ est suffisamment[†] petit,

l'algorithme converge vers un point $\beta^\star$ tel que $\nabla_\beta \ell\,(\beta^\star) = 0$.

[†] voir la page Wikipédia sur le théorème de Kantorovich pour une quantification du « suffisamment »

Maximisation de ℓ par la méthode de Newton-Raphson

Même algorithme mais cette fois en dimension $p+1$, avec :

- $\phi \to \nabla_\beta \ell$
- $\phi' \to \nabla^2_\beta \ell$

D'où la récurrence :

$$\beta^{(k+1)} = \beta^{(k)} - \left[\nabla^2_\beta \ell\left(\beta^{(k)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(k)}\right)$$

Sous les conditions suivantes :

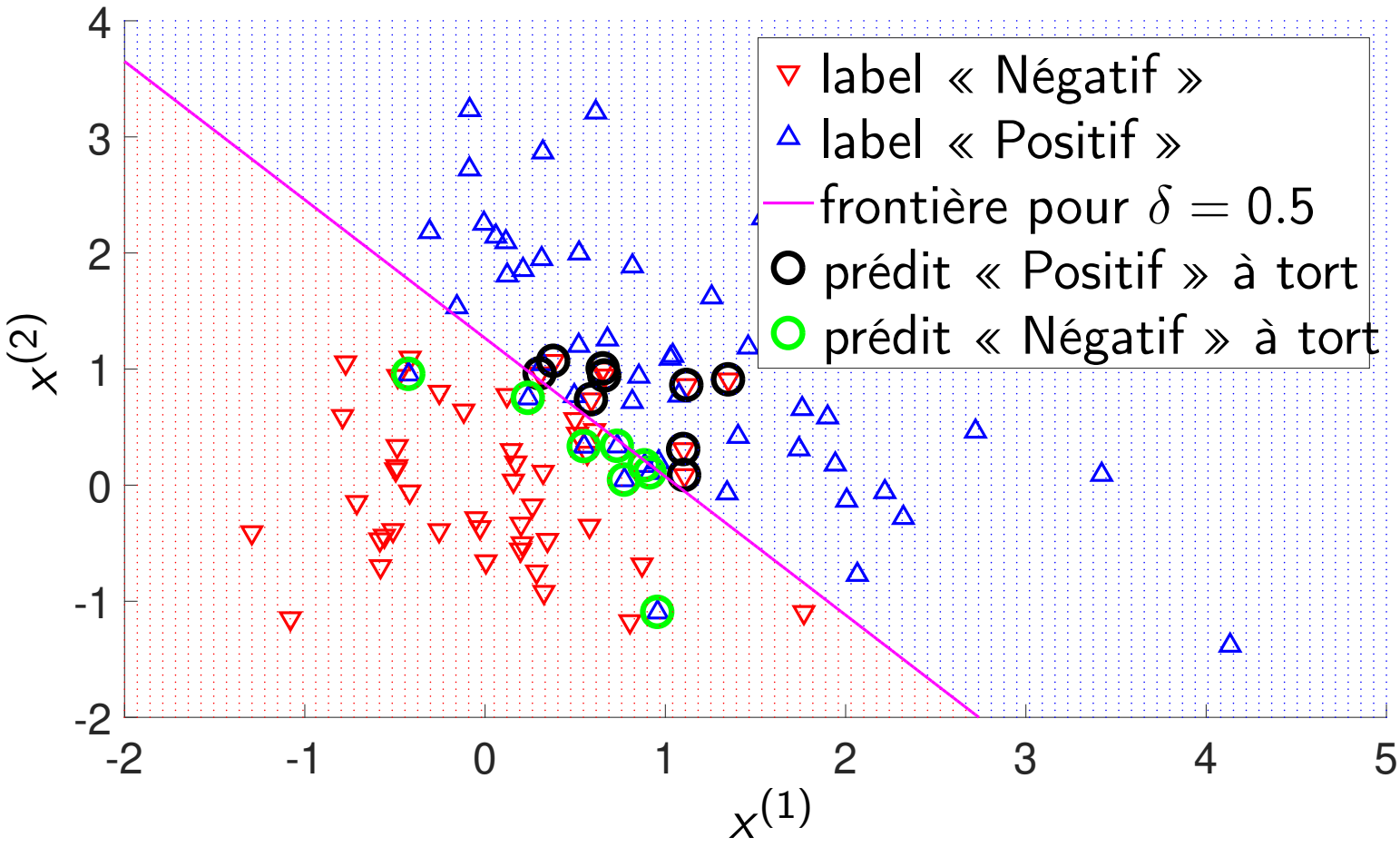
- $\nabla^2_\beta \ell\,(.)$ est une application lipschitzienne,
- $\nabla^2_\beta \ell\left(\beta^{(0)}\right)$ est inversible,
- $h_0 = \left[\nabla^2_\beta \ell\left(\beta^{(0)}\right)\right]^{-1} \nabla_\beta \ell\left(\beta^{(0)}\right)$ est suffisamment† petit,

l'algorithme converge vers un point $\beta^\star$ tel que $\nabla_\beta \ell\,(\beta^\star) = 0$.

† voir la page Wikipédia sur le théorème de Kantorovich pour une quantification du « suffisamment »

Plan du cours

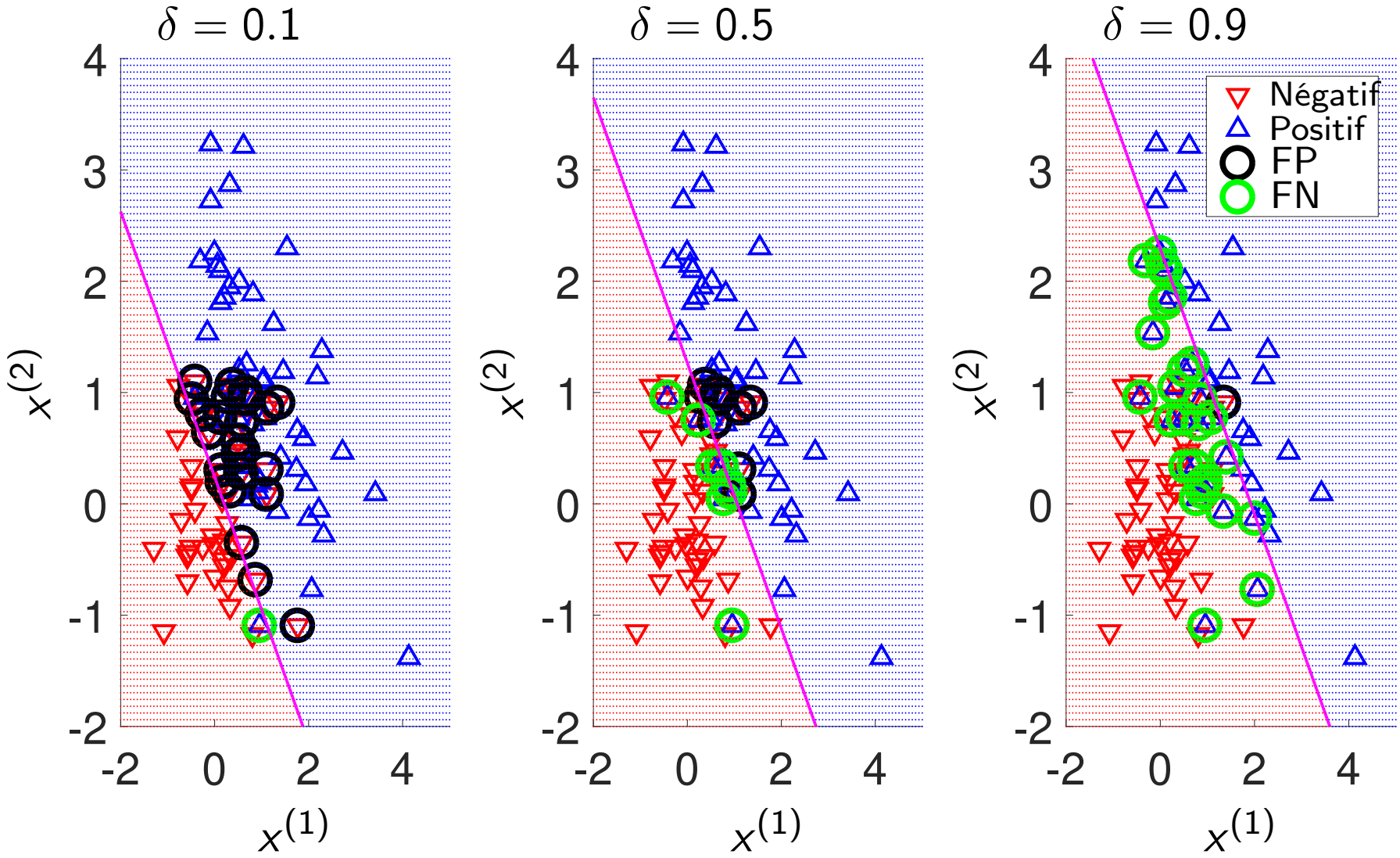
RL sur l'exemple avec 2 variables explicatives



Erreurs de prédiction :

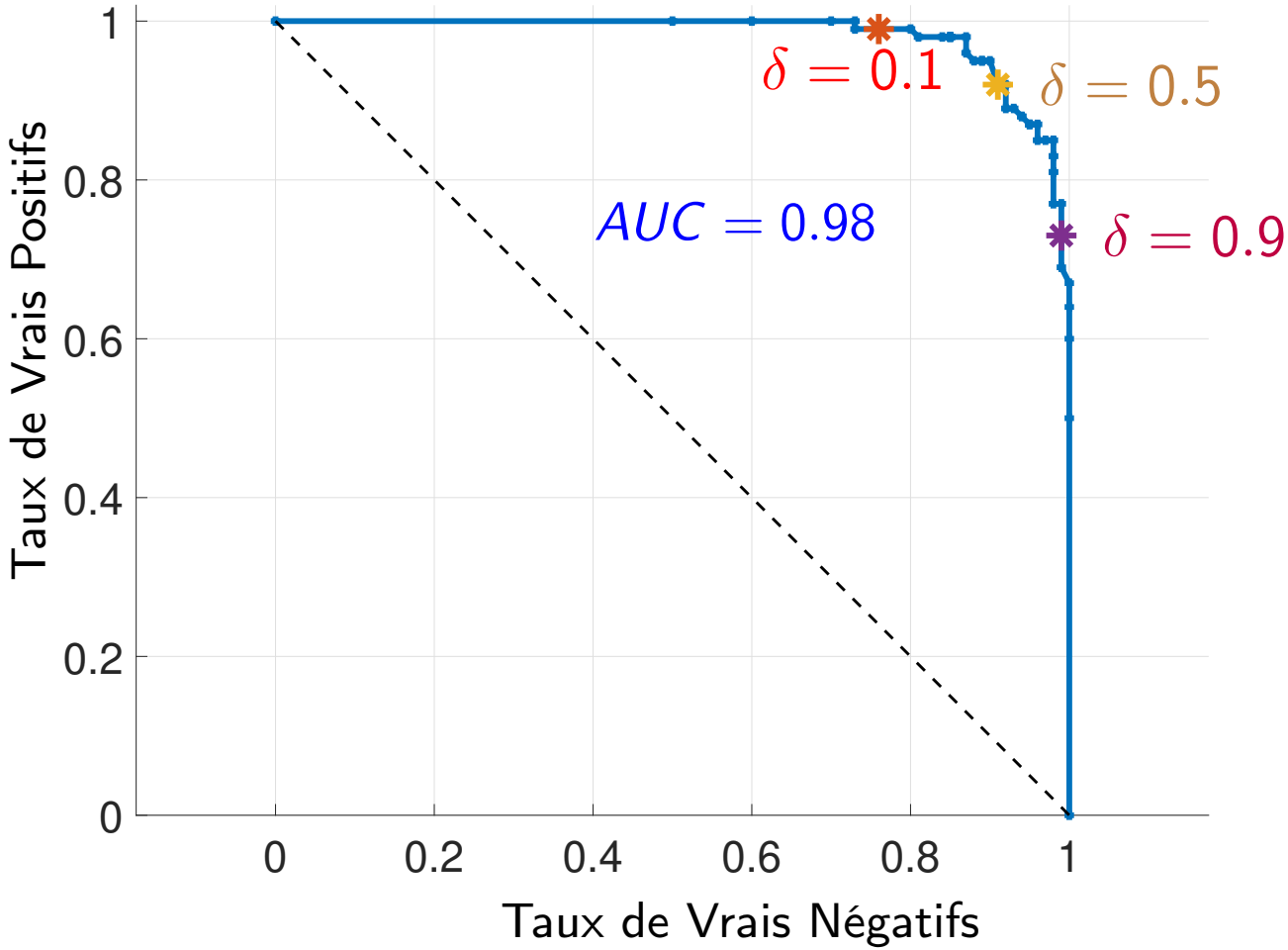
- prédire « Positif » un objet de classe « Négatif »
- prédire « Négatif » un objet de classe « Positif »

Influence de δ



Courbe ROC (Receiver Operating Characteristic)

- un outil d'aide à la décision (choix de δ)
- un outil de comparaison de classifieurs
- grandeur dérivée : *AUC* = Area Under the Curve



Plan du cours

Extension : grand nombre de variables

Gérer le cas où p est grand

On pénalise la log-vraisemblance :

- L_1 : $\hat{\beta} = \arg\max_\beta \left(\ell(\beta) - \lambda\|\beta\|^2\right)$
- L_2 : $\hat{\beta} = \arg\max_\beta \left(\ell(\beta) - \lambda\|\beta\|_1\right)$

➠ voir cours 8

p est « grand » si $p \gg n$, ou même simplement $p \approx n$

Extension : plus de deux classes

Classification multi-classes

Soit $\{0, 1, \ldots, K-1\}$ l'ensemble des labels (classes), $K \geq 3$.

On réalise $K-1$ régressions logistiques binaires en considérant une classe (ici « 0 ») comme référence :

$$\begin{cases} \ln\left(\frac{\mathrm{P}(Y=1|X=x)}{\mathrm{P}(Y=0|X=x)}\right) & = & \beta_{1,0} + \beta_1^\top x \\ \vdots & & \\ \ln\left(\frac{\mathrm{P}(Y=K-1|X=x)}{\mathrm{P}(Y=0|X=x)}\right) & = & \beta_{K-1,0} + \beta_{K-1}^\top x \end{cases}$$

Plan du cours

Plan du cours

Classification binaire : détection de spam

Données recueillies sur 4601 messages électroniques

- variables explicatives : fréquences relatives des 57 mots les plus fréquents
- variable à expliquer : étiquette « Spam » ou « Email »
  - ➠ variable qualitative (binaire ici)

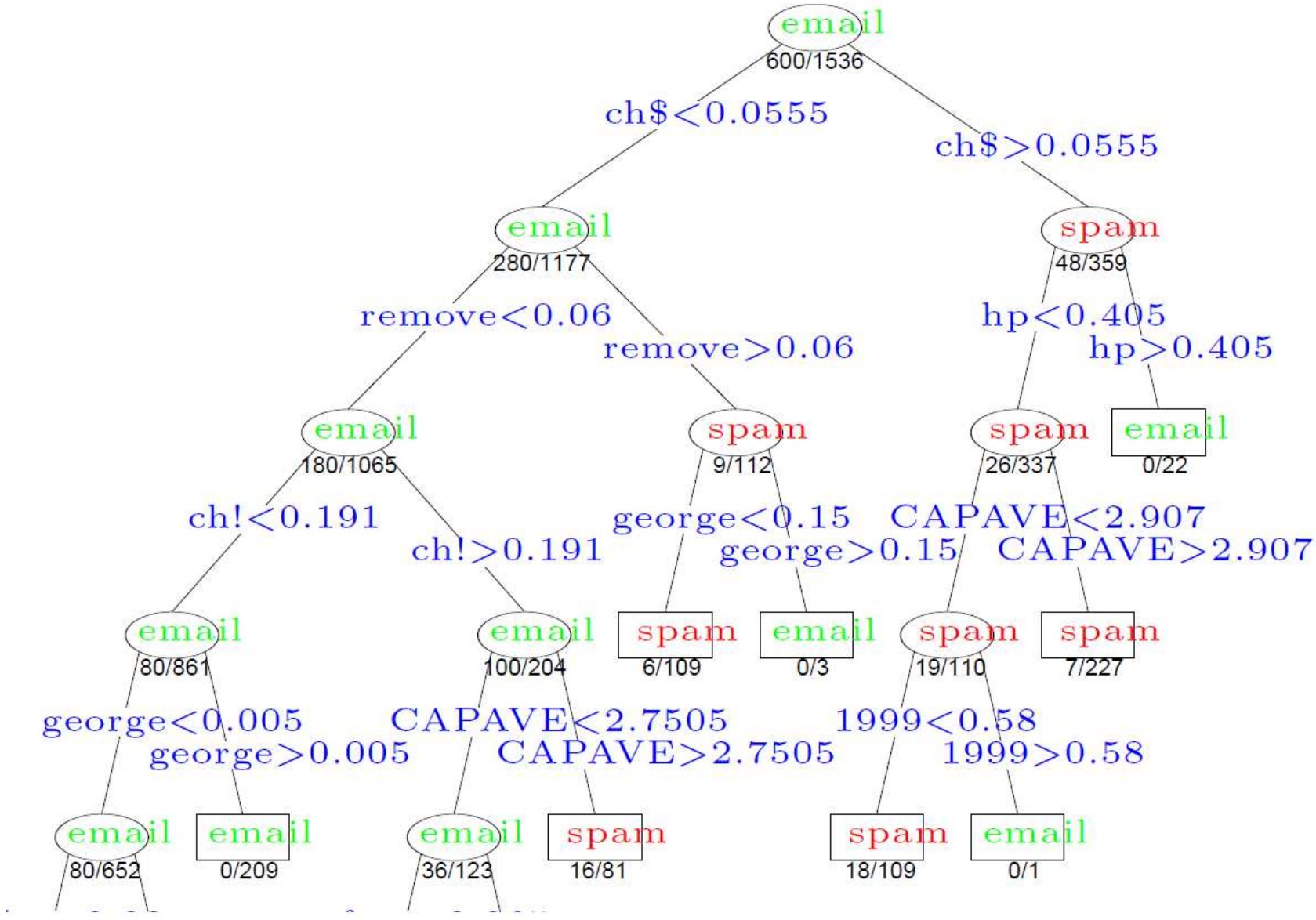
TABLE 1.1. *Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between* `spam` *and* `email`.

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

TABLE 9.3. *Spam data: confusion rates for the* 17*-node tree (chosen by cross–validation) on the test data. Overall error rate is* 9.3%.

	Predicted	
True	email	spam
email	57.3%	4.0%
spam	5.3%	33.4%

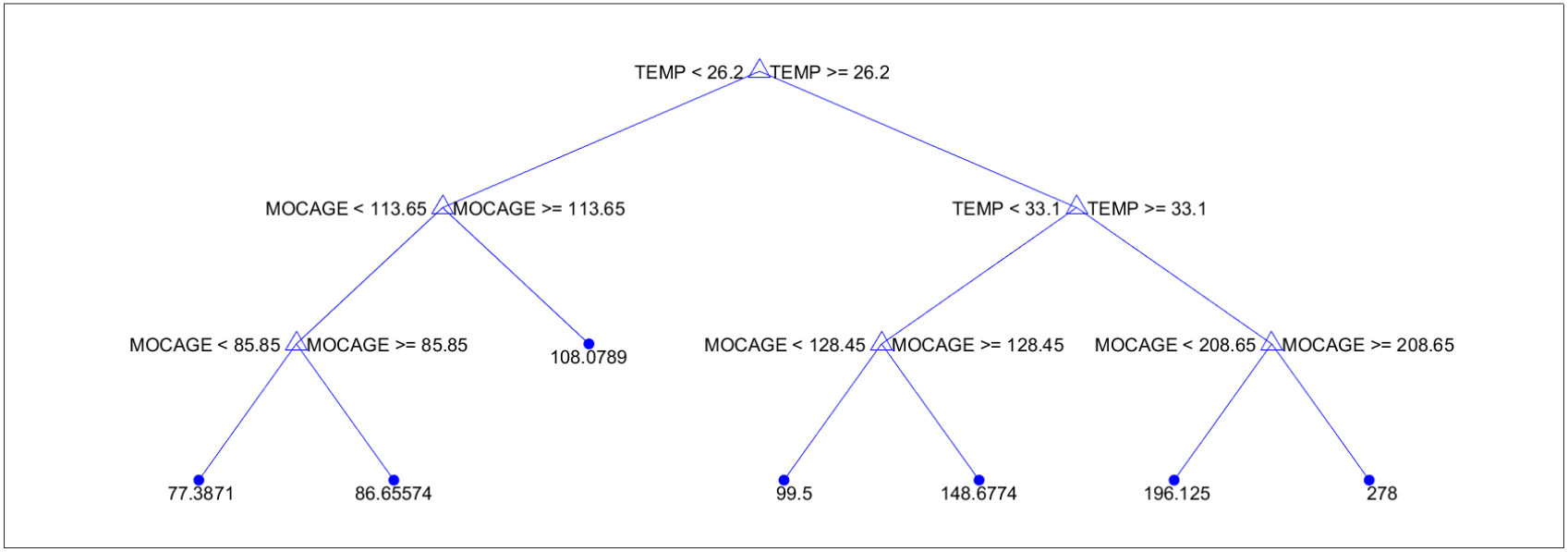
Source : The Elements of Statistical Learning, Springer (ainsi que la figure transparent suivant)


email
600/1536
ch$<0.0555
ch$>0.0555
email
280/1177
spam
48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email
180/1065
spam
9/112
spam
26/337
email
0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email
80/861
email
100/204
spam
6/109
email
0/3
spam
19/110
spam
7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email
80/652
email
0/209
email
36/123
spam
16/81
spam
18/109
email
0/1

Arbre de régression : exemple « Ozone »

Exemple simplifié (afin de permettre des visualisations 2D et 3D)

- prédire la variable 03 (variable quantitative)
- à partir des variables MOCAGE et TEMP



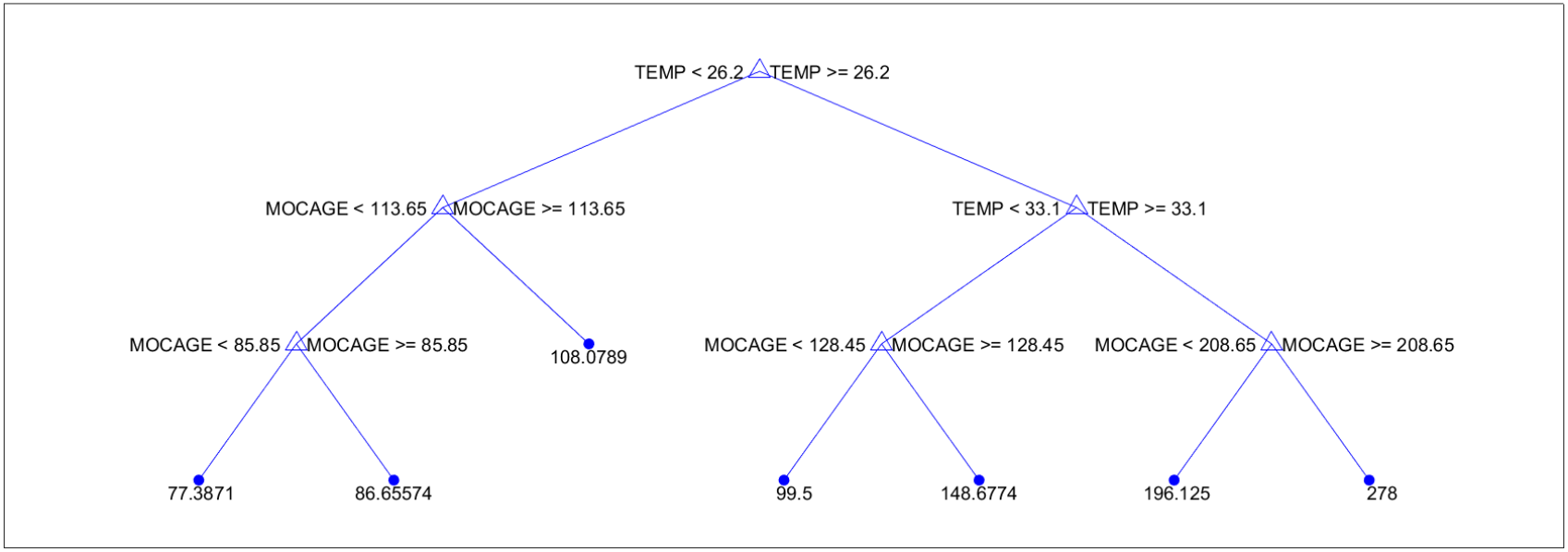
Vocabulaire. Quand la variable à prédire est

- quantitative, on parle d'arbre de régression
- qualitative, on parle d'arbre de classification

Arbre de régression : exemple « Ozone »

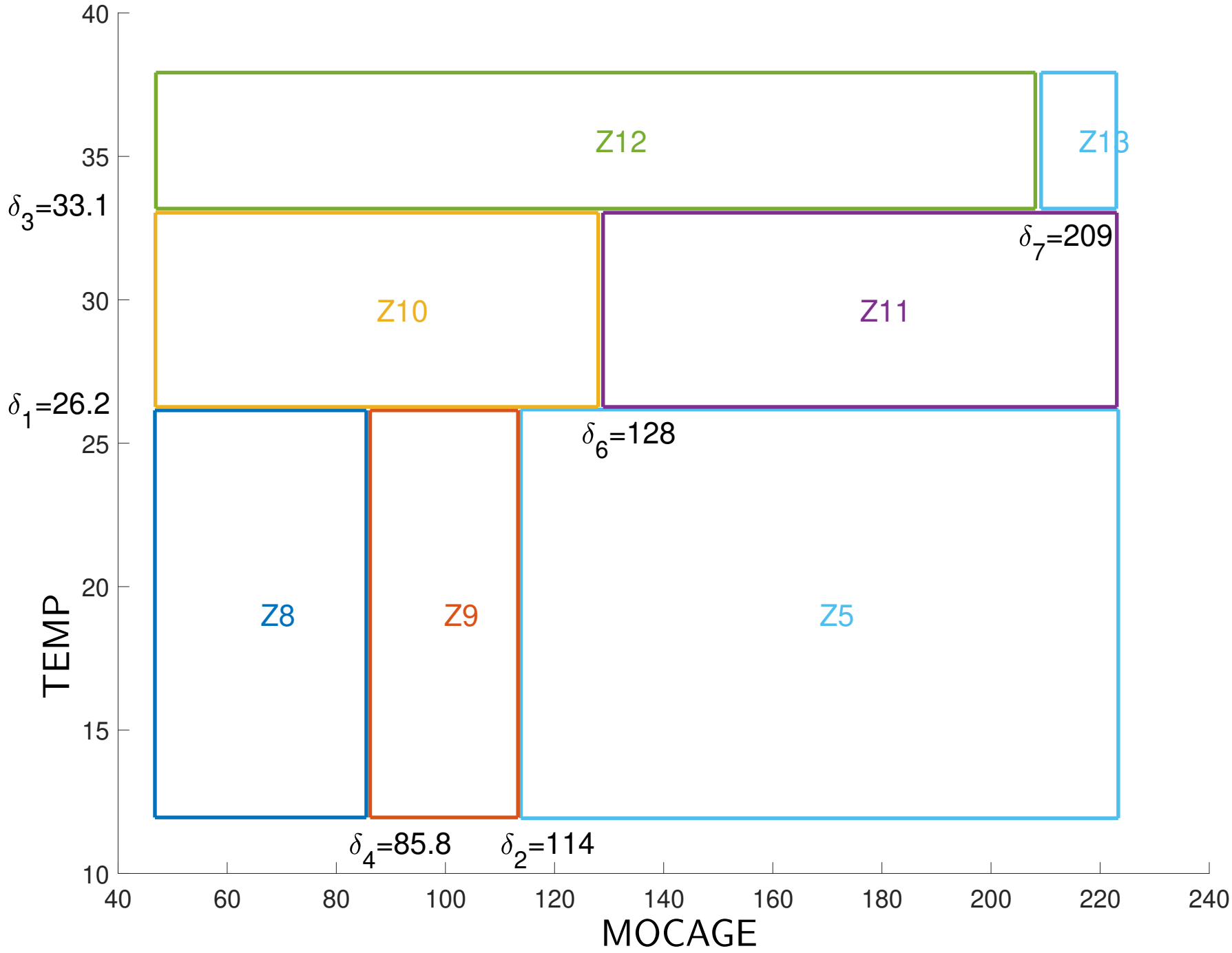
Exemple simplifié (afin de permettre des visualisations 2D et 3D)

- prédire la variable 03 (variable quantitative)
- à partir des variables MOCAGE et TEMP

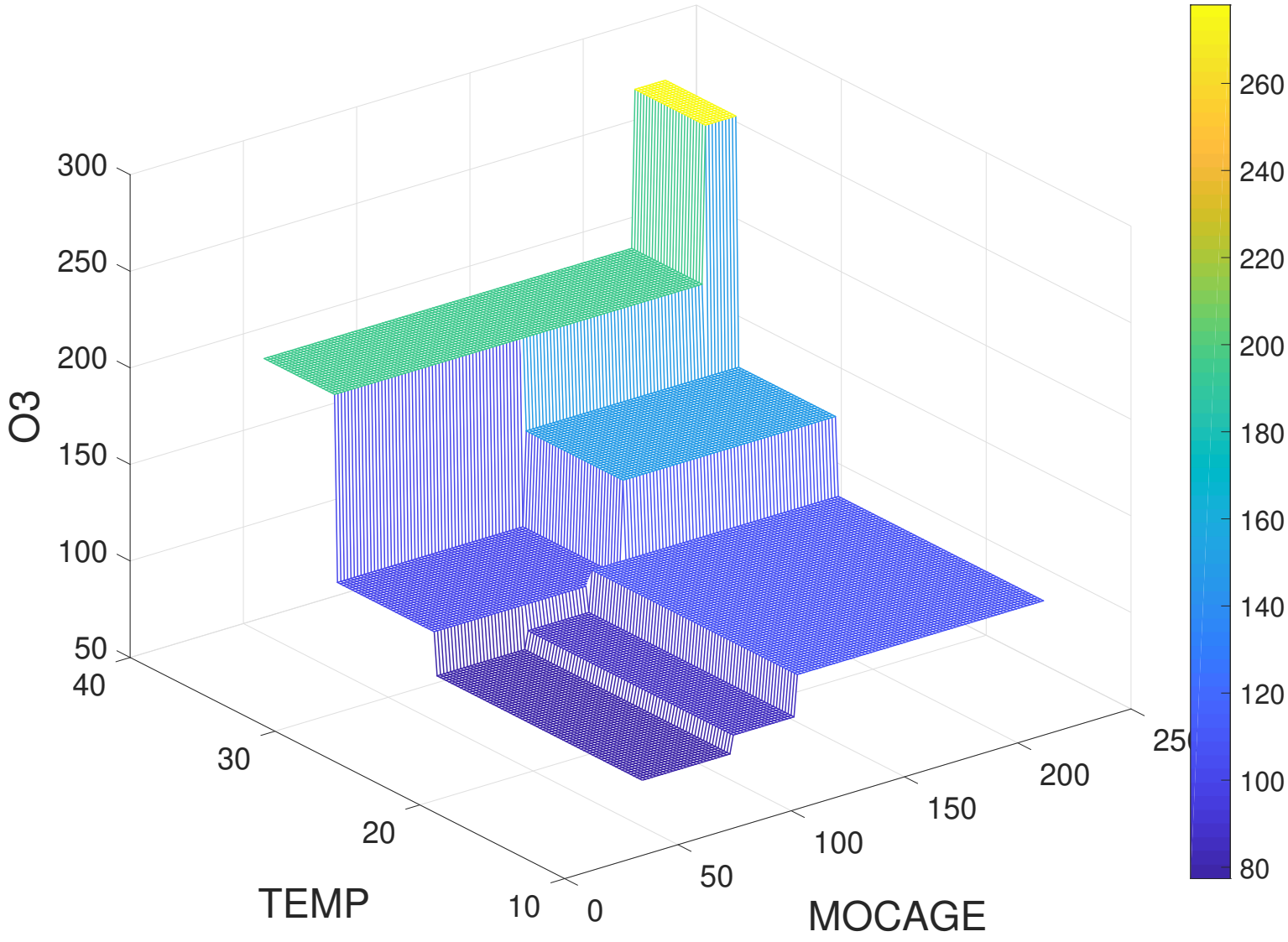


Vocabulaire. Quand la variable à prédire est

- quantitative, on parle d'arbre de régression
- qualitative, on parle d'arbre de classification


Z12
Z13
Z10
Z11
Z8
Z9
Z5
δ_3=33.1
δ_1=26.2
δ_7=209
δ_6=128
δ_4=85.8
δ_2=114
TEMP
MOCAGE
40
35
30
25
20
15
10
40
60
80
100
120
140
160
180
200
220
240

Arbre de régression : exemple « Ozone »



Plan du cours

Partitionnement récursif : principe général

Objectif

Construire une participation de $\mathcal{X}$ à partir des données $(\underline{X}, \underline{Y})$.

Principe : construction itérative d'une suite $(\mathcal{P}_m)_{m \geq 1}$ de partitions,

- $\mathcal{P}_m = \left\{ Z_1^{(m)}, \ldots, Z_m^{(m)} \right\}$, la partition $\mathcal{P}_m$ contenant m parties.

Initialisation : $\mathcal{P}_1 = \{\mathcal{X}\}$.

$\mathcal{P}_m \to \mathcal{P}_{m+1}$: on divise une partie $Z_{k_m}^{(m)}$ selon une des variables :

- $\tilde{Z}_1 = Z_{k_m}^{(m)} \cap \left\{ x \text{ tel que } x^{(j_m)} \leq \delta_m \right\}$
- $\tilde{Z}_2 = Z_{k_m}^{(m)} \cap \left\{ x \text{ tel que } x^{(j_m)} > \delta_m \right\}$

(l'indice j_m et le seuil δ_m restent à préciser)

Partitionnement récursif : principe général

Objectif

Construire une participation de $\mathcal{X}$ à partir des données $(\underline{X}, \underline{Y})$.

Principe : construction itérative d'une suite $(\mathcal{P}_m)_{m \geq 1}$ de partitions,

- $\mathcal{P}_m = \left\{ Z_1^{(m)}, \ldots, Z_m^{(m)} \right\}$, la partition $\mathcal{P}_m$ contenant m parties.

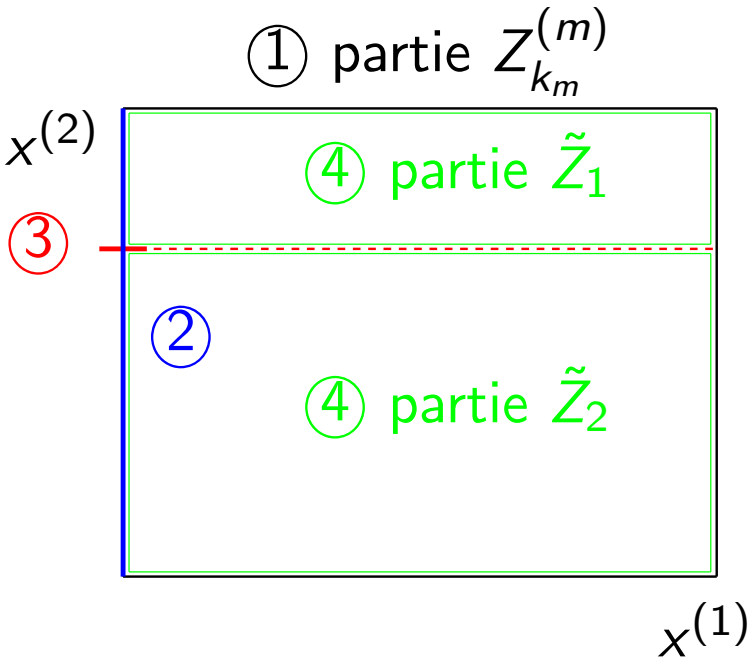
Initialisation : $\mathcal{P}_1 = \{\mathcal{X}\}$.

$\mathcal{P}_m \to \mathcal{P}_{m+1}$: on divise une partie $Z_{k_m}^{(m)}$ selon une des variables :

- $\tilde{Z}_1 = Z_{k_m}^{(m)} \cap \left\{ x \text{ tel que } x^{(j_m)} \leq \delta_m \right\}$
- $\tilde{Z}_2 = Z_{k_m}^{(m)} \cap \left\{ x \text{ tel que } x^{(j_m)} > \delta_m \right\}$

(l'indice j_m et le seuil δ_m restent à préciser)

Exemple avec $p = 2$



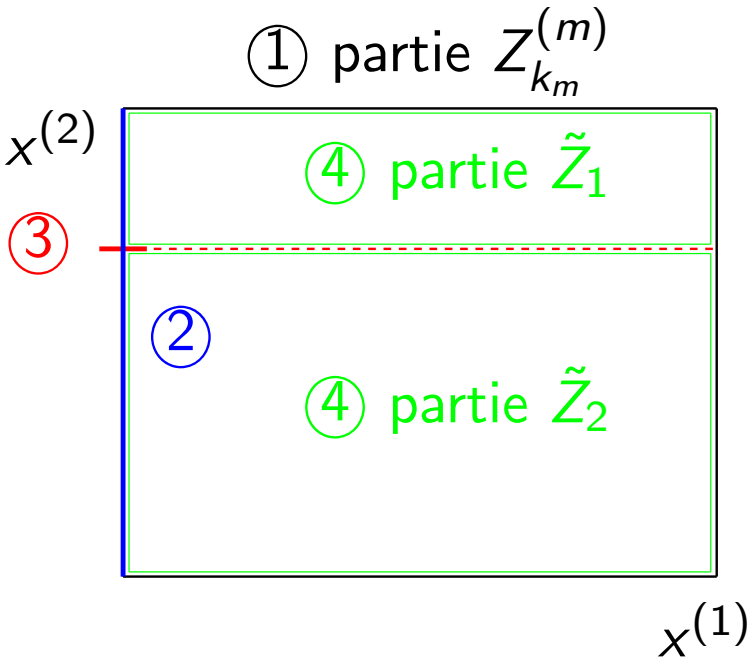
Itération $\mathcal{P}_m \to \mathcal{P}_{m+1}$:

- ① partie $Z_{k_m}^{(m)} \in \mathcal{P}_m$
- ② variable $x^{(j_m)}$ (ici $j_m = 2$)
- ③ seuil δ_m
- ④ création de $\tilde{Z}_1$ et $\tilde{Z}_2$

Après division de $Z_{k_m}^{(m)}$, on obtient :

$$\mathcal{P}_{m+1} = \mathcal{P}_m \bigcup \left\{ \tilde{Z}_1, \tilde{Z}_2 \right\} \setminus \left\{ Z_{k_m}^{(m)} \right\}$$

Exemple avec $p = 2$



Itération $\mathcal{P}_m \to \mathcal{P}_{m+1}$:

- ① partie $Z_{k_m}^{(m)} \in \mathcal{P}_m$
- ② variable $x^{(j_m)}$ (ici $j_m = 2$)
- ③ seuil δ_m
- ④ création de $\tilde{Z}_1$ et $\tilde{Z}_2$

Après division de $Z_{k_m}^{(m)}$, on obtient :

$$\mathcal{P}_{m+1} = \mathcal{P}_m \bigcup \left\{ \tilde{Z}_1, \tilde{Z}_2 \right\} \setminus \left\{ Z_{k_m}^{(m)} \right\}$$

Choisir k_m, j_m et δ_m

Soit $D(Z)$ une mesure de l'hétérogénéité de la partie Z.

Exemple (pour y quantitative)

$$D(Z) = \sum_{i \in Z} (y_i - \bar{y}_Z)^2$$

où $\bar{y}_Z$ est la moyenne empirique calculée sur la partie Z.

k_m, j_m et δ_m sont choisis conjointement de sorte que

$D\left(Z_{k_m}^{(m)}\right) - D(\tilde{Z}_1) - D(\tilde{Z}_2)$ soit le plus grand possible

➠ plus grande décroissance de l'hétérogénéité

(Rappel : $\tilde{Z}_1$ et $\tilde{Z}_2$ sont les deux parties créées par division de $Z_{k_m}^{(m)}$)

Choisir k_m, j_m et δ_m

Soit $D(Z)$ une mesure de l'hétérogénéité de la partie Z.

Exemple (pour y quantitative)

$$D(Z) = \sum_{i \in Z} (y_i - \bar{y}_Z)^2$$

où $\bar{y}_Z$ est la moyenne empirique calculée sur la partie Z.

k_m, j_m et δ_m sont choisis conjointement de sorte que

$D\left(Z_{k_m}^{(m)}\right) - D(\tilde{Z}_1) - D(\tilde{Z}_2)$ soit le plus grand possible

⇛ plus grande décroissance de l'hétérogénéité

(Rappel : $\tilde{Z}_1$ et $\tilde{Z}_2$ sont les deux parties créées par division de $Z_{k_m}^{(m)}$)

Plan du cours

Fonction de prédiction constante par morceaux

Les arbres de décisions réalisent une prédiction constante par morceaux sur les éléments de la partition :

$$h_\beta(x) = \sum_{k=1}^{m} \beta_k \mathbb{1}_{Z_k^{(m)}}(x).$$

Remarque : une fois la partition fixée, il s'agit d'un modèle linéaire par rapport aux m variables $\mathbb{1}_{Z_k^{(m)}}(x)$.

Fonction de prédiction constante par morceaux

Les arbres de décisions réalisent une prédiction constante par morceaux sur les éléments de la partition :

$$h_\beta(x) \;=\; \sum_{k=1}^{m} \beta_k \mathbb{1}_{Z_k^{(m)}}(x).$$

Remarque : une fois la partition fixée, il s'agit d'un modèle linéaire par rapport aux m variables $\mathbb{1}_{Z_k^{(m)}}(x)$.

Estimation des coefficients

Principe : pour estimer $\beta^{(m)} = \left(\beta_1^{(m)}, \dots \beta_m^{(m)}\right)$,

- on se donne une fonction de perte $L(y, h_\beta(x))$,
- puis on minimise le risque empirique.

Simplification :

$$\begin{aligned} \min_\beta \hat{\mathscr{R}}(h_\beta) &= \min_\beta \sum_{i=1}^n L(y_i, h_\beta(x_i)) \\ &= \min_\beta \sum_{k=1}^m \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \\ &= \sum_{k=1}^m \min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \end{aligned}$$

Conséquence : $\forall k,\ \hat{\beta}_k^{(m)} = \arg\min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k)$.

Estimation des coefficients

Principe : pour estimer $\beta^{(m)} = \left(\beta_1^{(m)}, \dots \beta_m^{(m)}\right)$,

- on se donne une fonction de perte $L(y, h_\beta(x))$,
- puis on minimise le risque empirique.

Simplification :

$$\begin{aligned} \min_\beta \hat{\mathscr{R}}(h_\beta) &= \min_\beta \sum_{i=1}^n L(y_i, h_\beta(x_i)) \\ &= \min_\beta \sum_{k=1}^m \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \\ &= \sum_{k=1}^m \min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \end{aligned}$$

Conséquence : $\forall k,\ \hat{\beta}_k^{(m)} = \arg\min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k)$.

Estimation des coefficients

Principe : pour estimer $\beta^{(m)} = \left(\beta_1^{(m)}, \dots \beta_m^{(m)}\right)$,

- on se donne une fonction de perte $L(y, h_\beta(x))$,
- puis on minimise le risque empirique.

Simplification :

$$\begin{aligned} \min_\beta \hat{\mathscr{R}}(h_\beta) &= \min_\beta \sum_{i=1}^n L(y_i, h_\beta(x_i)) \\ &= \min_\beta \sum_{k=1}^m \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \\ &= \sum_{k=1}^m \min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k) \end{aligned}$$

Conséquence : $\forall k,\ \hat{\beta}_k^{(m)} = \arg\min_{\beta_k} \sum_{i \in Z_k^{(m)}} L(y_i, \beta_k)$.

Deux cas particuliers importants

Régression avec la perte quadratique

$$\hat{\beta}_k^{(m)} = \text{argmin}_{\beta_k} \sum_{i \in Z_k^{(m)}} (y_i - \beta_k)^2 = \overline{y}_{Z_k^{(m)}}$$

Classification binaire avec la perte logarithmique

Classification douce :

$$\hat{\beta}_k^{(m)} = \text{argmin}_{\beta_k \in [0,1]} \sum_{i \in Z_k^{(m)}} (-y_i \ln(\beta_k) - (1 - y_i) \ln(1 - \beta_k))$$

$$= \frac{1}{\text{card}\left(Z_k^{(m)}\right)} \cdot \text{card}\left(i \in Z_k^{(m)} \text{ tel que } y_i = 1\right)$$

Classification dure : seuiller avec $\delta = \frac{1}{2}$ (cf. régression logistique).

Deux cas particuliers importants

Régression avec la perte quadratique

$$\hat{\beta}_k^{(m)} = \text{argmin}_{\beta_k} \sum_{i \in Z_k^{(m)}} (y_i - \beta_k)^2 = \overline{y}_{Z_k^{(m)}}$$

Classification binaire avec la perte logarithmique

Classification douce :

$$\hat{\beta}_k^{(m)} = \text{argmin}_{\beta_k \in [0,1]} \sum_{i \in Z_k^{(m)}} (-y_i \ln(\beta_k) - (1 - y_i) \ln(1 - \beta_k))$$

$$= \frac{1}{\text{card}\left(Z_k^{(m)}\right)} \cdot \text{card}\left(i \in Z_k^{(m)} \text{ tel que } y_i = 1\right)$$

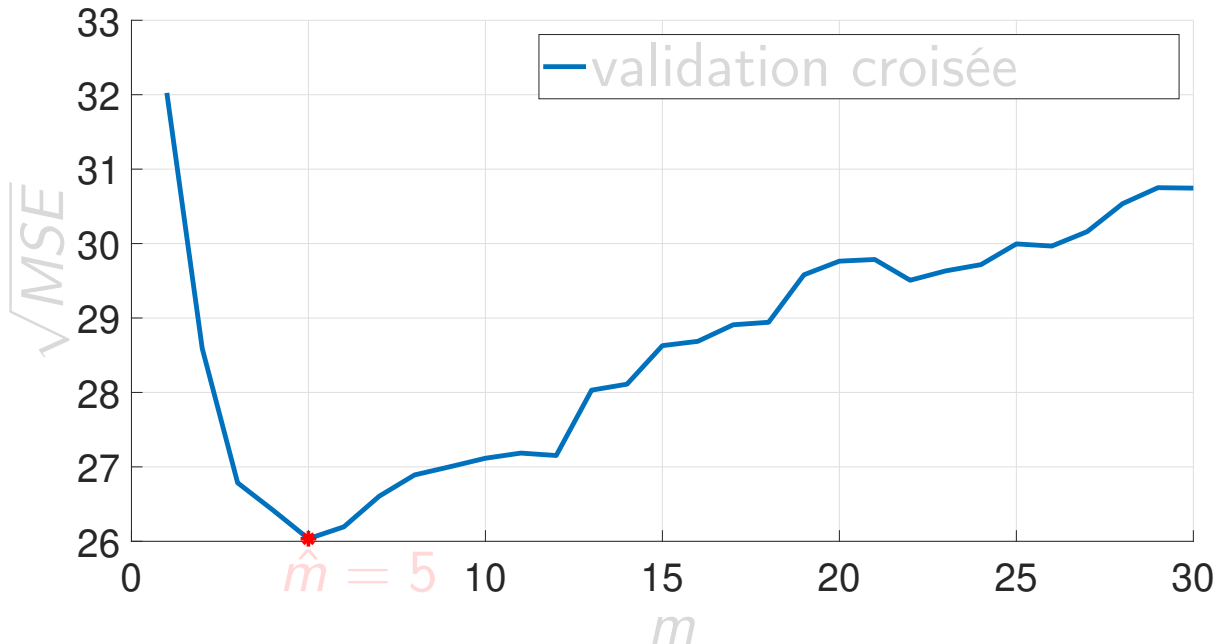
Classification dure : seuiller avec $\delta = \frac{1}{2}$ (cf. régression logistique).

Choix de la taille m de la partition

- m peut être fixé à l'avance (a priori sur la complexité désirée)
- ou estimé par validation croisée.

Exemple « Ozone »

- Régression de la variable 03 à partir des 7 variables explicatives
- Choix de m par validation croisée leave-one-out

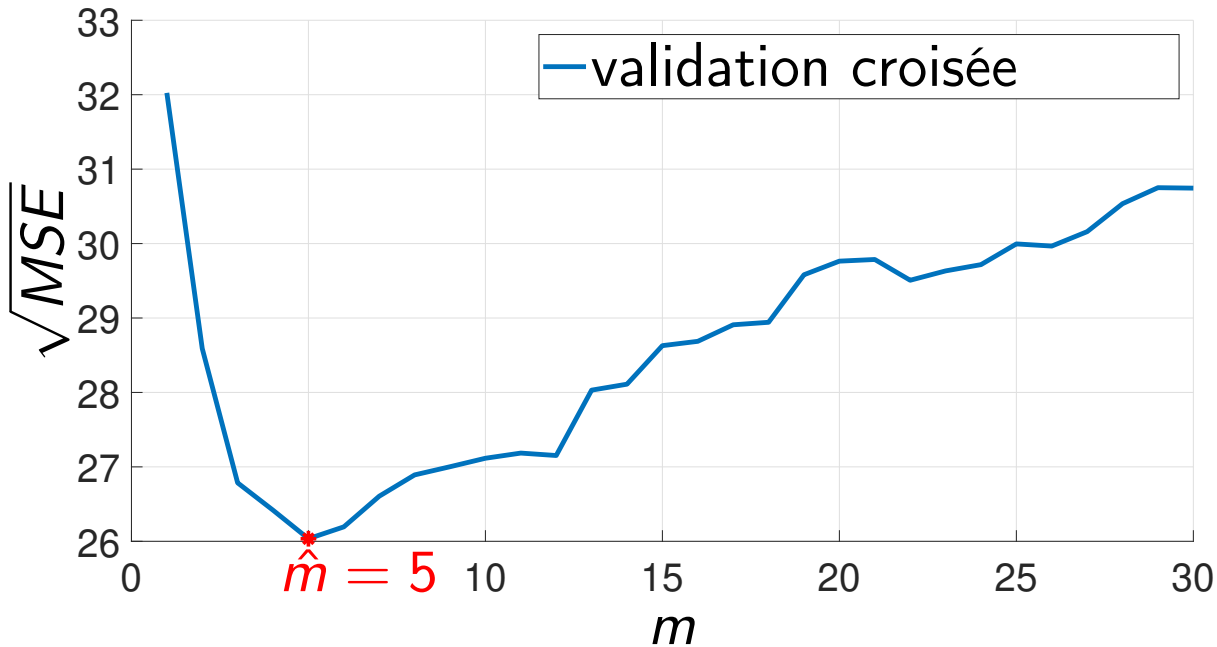


Choix de la taille m de la partition

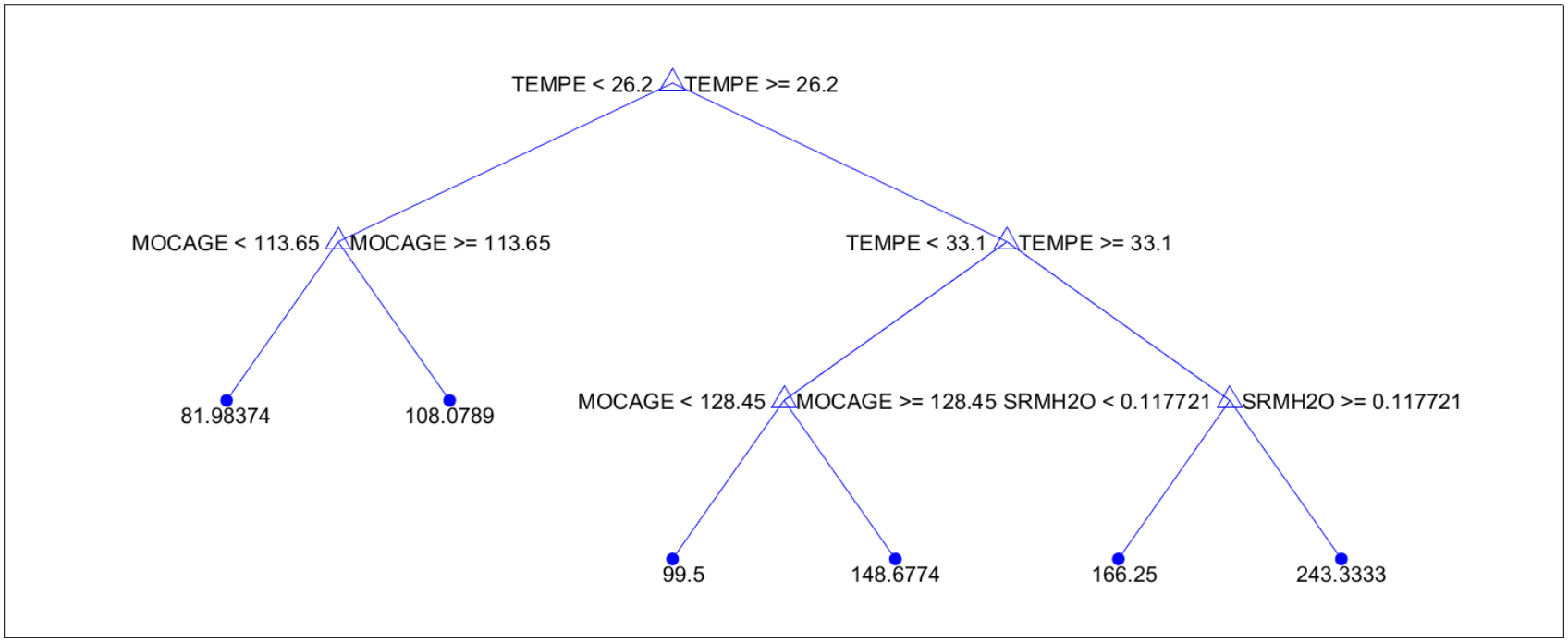
- m peut être fixé à l'avance (a priori sur la complexité désirée)
- ou estimé par validation croisée.

Exemple « Ozone »

- Régression de la variable 03 à partir des 7 variables explicatives
- Choix de m par validation croisée leave-one-out



Arbre de régression : exemple « Ozone »




```
 1  if TEMPE<26.2 then node 2 elseif TEMPE>=26.2 then node 3 else 103.433
 2  if MOCAGE<113.65 then node 4 elseif MOCAGE>=113.65 then node 5 else 88.1429
 3  if TEMPE<33.1 then node 6 elseif TEMPE>=33.1 then node 7 else 153.673
 4  fit = 81.9837
 5  fit = 108.079
 6  if MOCAGE<128.45 then node 8 elseif MOCAGE>=128.45 then node 9 else 138.59
 7  if SRMH2O<0.117721 then node 10 elseif SRMH2O>=0.117721 then node 11 else 212.5
 8  fit = 99.5
 9  fit = 148.677
10  fit = 166.25
11  fit = 243.333
```

Pour aller plus loin...

Défauts des arbres de décision

- grande sensibilité aux fluctuations de l'échantillon $(\underline{x}, \underline{y})$
- réponse constante sur chaque partie (par définition)
  (y compris lorsque le phénomène à modéliser est régulier)

Extensions

- agrégation de modèles (chacun étant un arbre)
  - ➠ Forêts aléatoires
- somme pondérée de classifieurs faibles
  - ➠ Boosting (AdaBoost)

Pour aller plus loin...

Défauts des arbres de décision

- grande sensibilité aux fluctuations de l'échantillon $(\underline{x}, \underline{y})$
- réponse constante sur chaque partie (par définition)
  (y compris lorsque le phénomène à modéliser est régulier)

Extensions

- agrégation de modèles (chacun étant un arbre)
  - ➠ Forêts aléatoires
- somme pondérée de classifieurs faibles
  - ➠ Boosting (AdaBoost)

Plan du cours

Plan du cours

Le neurone biologique (multipolaire) : axone, dendrites...

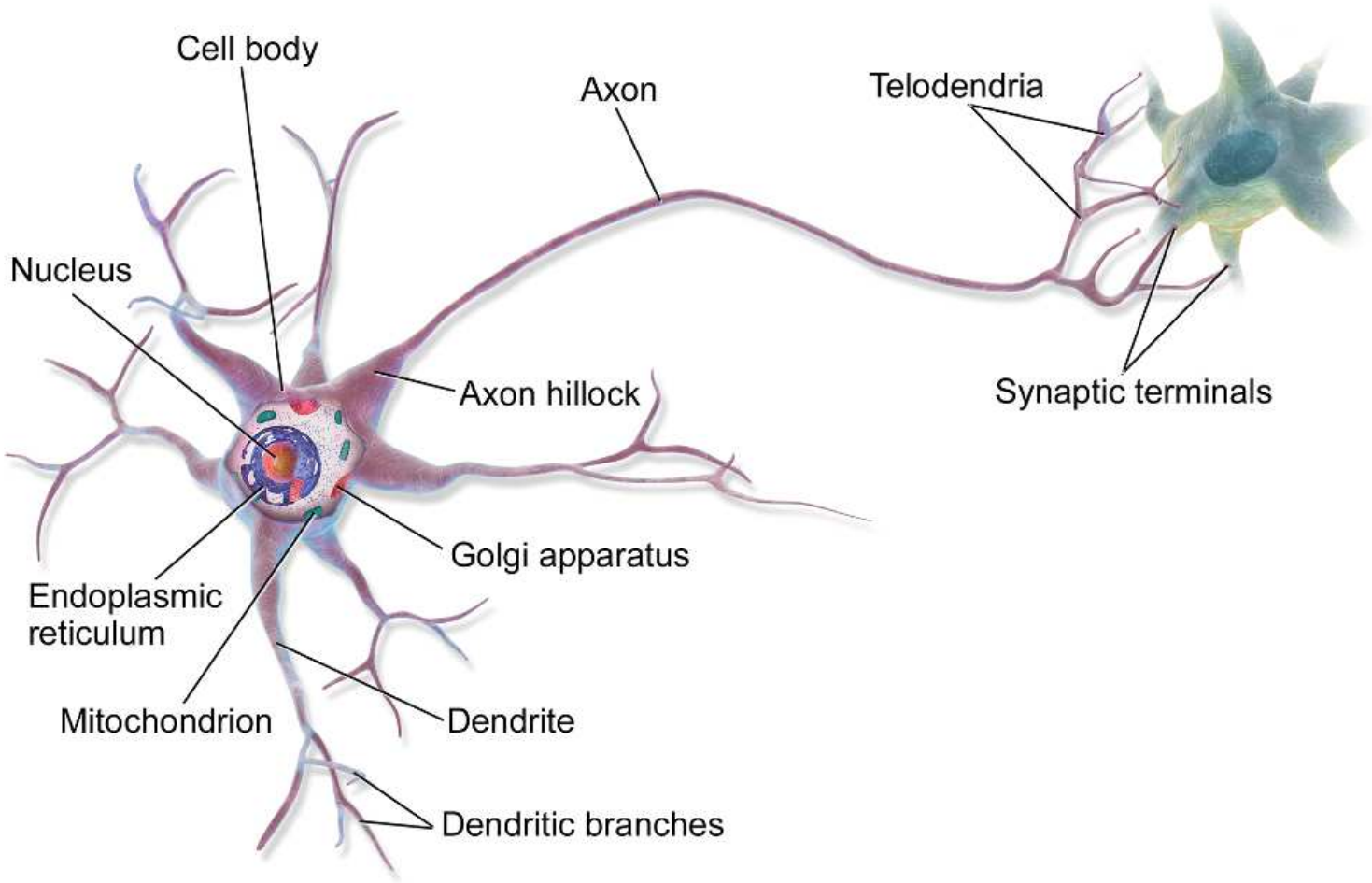


Image : Bruce Blaus, `https://commons.wikimedia.org`, CC BY 3.0

« Un neurone multipolaire est un type de neurone qui possède un seul (généralement long) axone et de nombreux dendrites, permettant l'intégration d'un grand nombre d'informations provenant d'autres neurones. » (`https://fr.wikipedia.org/wiki/Neurone_multipolaire`)

Le neurone artificiel

Définition : neurone (McCulloch and Pitts, 1943)[†]

En apprentissage statistique, un neurone à p variables (entrées) est une fonction, en général non-linéaire[‡], de la forme

$$h(x) = \varphi(w\,x + b), \quad x \in \mathbb{R}^p,$$

où

- φ est une fonction $\mathbb{R} \to \mathbb{R}$ croissante ;
- $w \in \mathbb{R}^{1\times p}$, et $b \in \mathbb{R}$.

Vocabulaire

- φ : fonction d'activation,
- $w_1, \ldots, w_p$: poids,
- b : biais (rien à avoir avec le biais d'un estimateur).

[†] Le neurone de McCulloch & Pitts (1943) utilisait spécifiquement $\varphi = \mathrm{sgn}$ comme fonction d'activation.

[‡] On verra plus loin une situation où l'on fait intervenir un neurone linéaire ($\varphi = \mathrm{Id}$).

Le neurone artificiel

Définition : neurone (McCulloch and Pitts, 1943)[†]

En apprentissage statistique, un neurone à p variables (entrées) est une fonction, en général non-linéaire[‡], de la forme

$$h(x) = \varphi(w\,x + b), \quad x \in \mathbb{R}^p,$$

où

- φ est une fonction $\mathbb{R} \to \mathbb{R}$ croissante ;
- $w \in \mathbb{R}^{1\times p}$, et $b \in \mathbb{R}$.

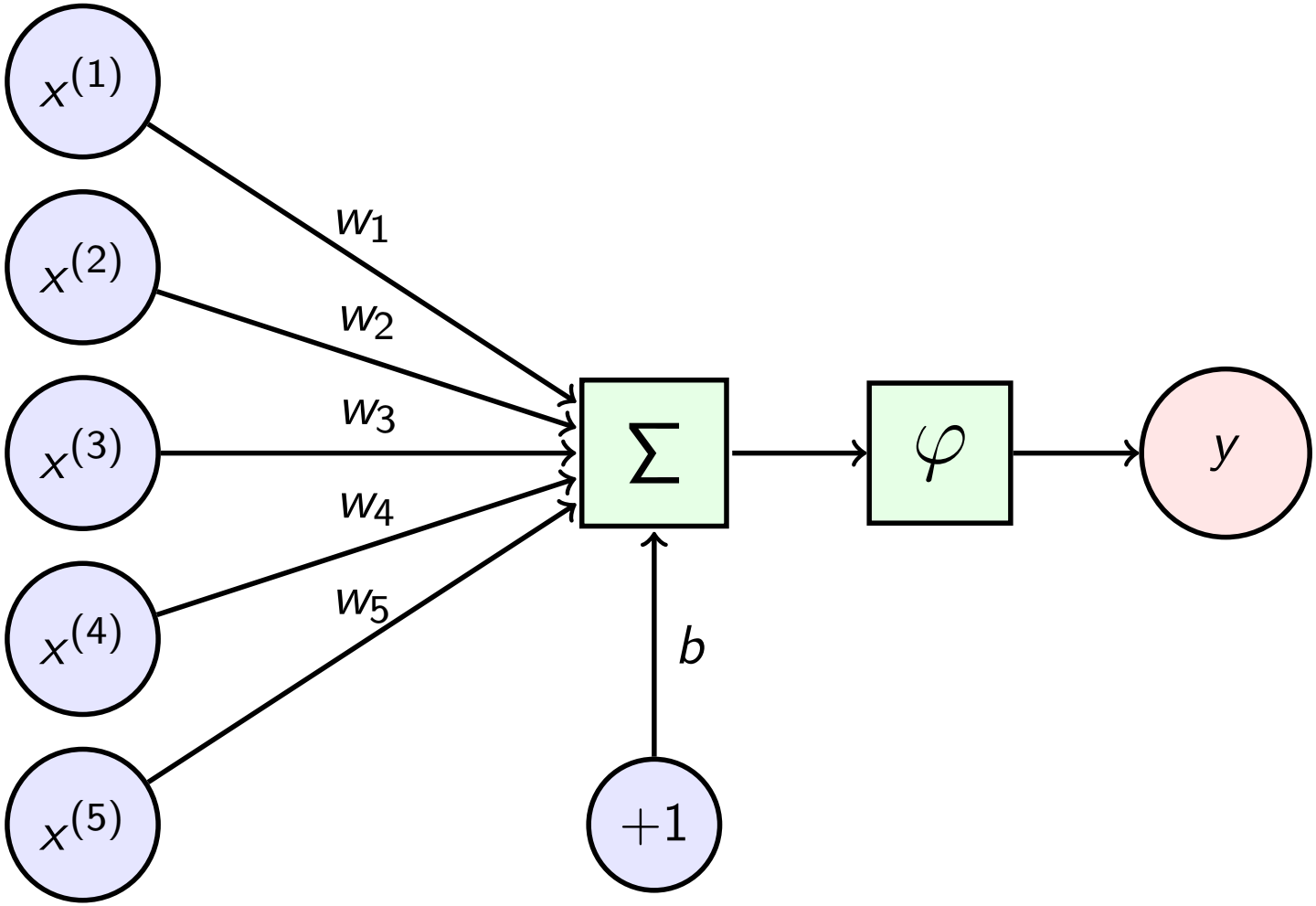
Vocabulaire

- φ : fonction d'activation,
- $w_1, \ldots, w_p$: poids,
- b : biais (rien à avoir avec le biais d'un estimateur).

[†] Le neurone de McCulloch & Pitts (1943) utilisait spécifiquement $\varphi = \mathrm{sgn}$ comme fonction d'activation.

[‡] On verra plus loin une situation où l'on fait intervenir un neurone linéaire ($\varphi = \mathrm{Id}$).

Le neurone artificiel : illustration ($p = 5$)



Fonctions d'activation

Fonctions d'activation discontinues (non recommandées[†]) :

- fonction de Heaviside : $\varphi(v) = \mathbb{1}_{v \geq 0}$, ou
- fonction signe : $\varphi(v) = \mathrm{sgn}(v) = \mathbb{1}_{v>0} - \mathbb{1}_{v<0}$.

Fonctions en « forme de S », ou sigmoïdes :

- logistique[‡] : $\varphi(v) = \frac{1}{1+e^{-v}} = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{v}{2}\right)$, ou
- tanh : $\varphi(v) = \tanh(v) = \frac{e^{v}-e^{-v}}{e^{v}+e^{-v}}$.

Fonction ReLU (*Rectified Linear Unit*) :

- $\varphi(v) = \max(0, v)$.

[†] Utilisée dans les modèles les plus anciens, notamment le perceptron de Rosenblatt (1957), mais abandonnées depuis pour cause de dérivée nulle pp, qui rend difficile la minimisation du risque.

[‡] Le terme « sigmoïde » désigne parfois cette fonction particulière.

Fonctions d'activation

Fonctions d'activation discontinues (non recommandées[†]) :

- fonction de Heaviside : $\varphi(v) = \mathbb{1}_{v \geq 0}$, ou
- fonction signe : $\varphi(v) = \operatorname{sgn}(v) = \mathbb{1}_{v>0} - \mathbb{1}_{v<0}$.

Fonctions en « forme de S », ou sigmoïdes :

- logistique[‡] : $\varphi(v) = \frac{1}{1+e^{-v}} = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{v}{2}\right)$, ou
- tanh : $\varphi(v) = \tanh(v) = \frac{e^{v}-e^{-v}}{e^{v}+e^{-v}}$.

Fonction ReLU (*Rectified Linear Unit*) :

- $\varphi(v) = \max(0, v)$.

[†] Utilisée dans les modèles les plus anciens, notamment le perceptron de Rosenblatt (1957), mais abandonnées depuis pour cause de dérivée nulle pp, qui rend difficile la minimisation du risque.

[‡] Le terme « sigmoïde » désigne parfois cette fonction particulière.

Fonctions d'activation (suite)

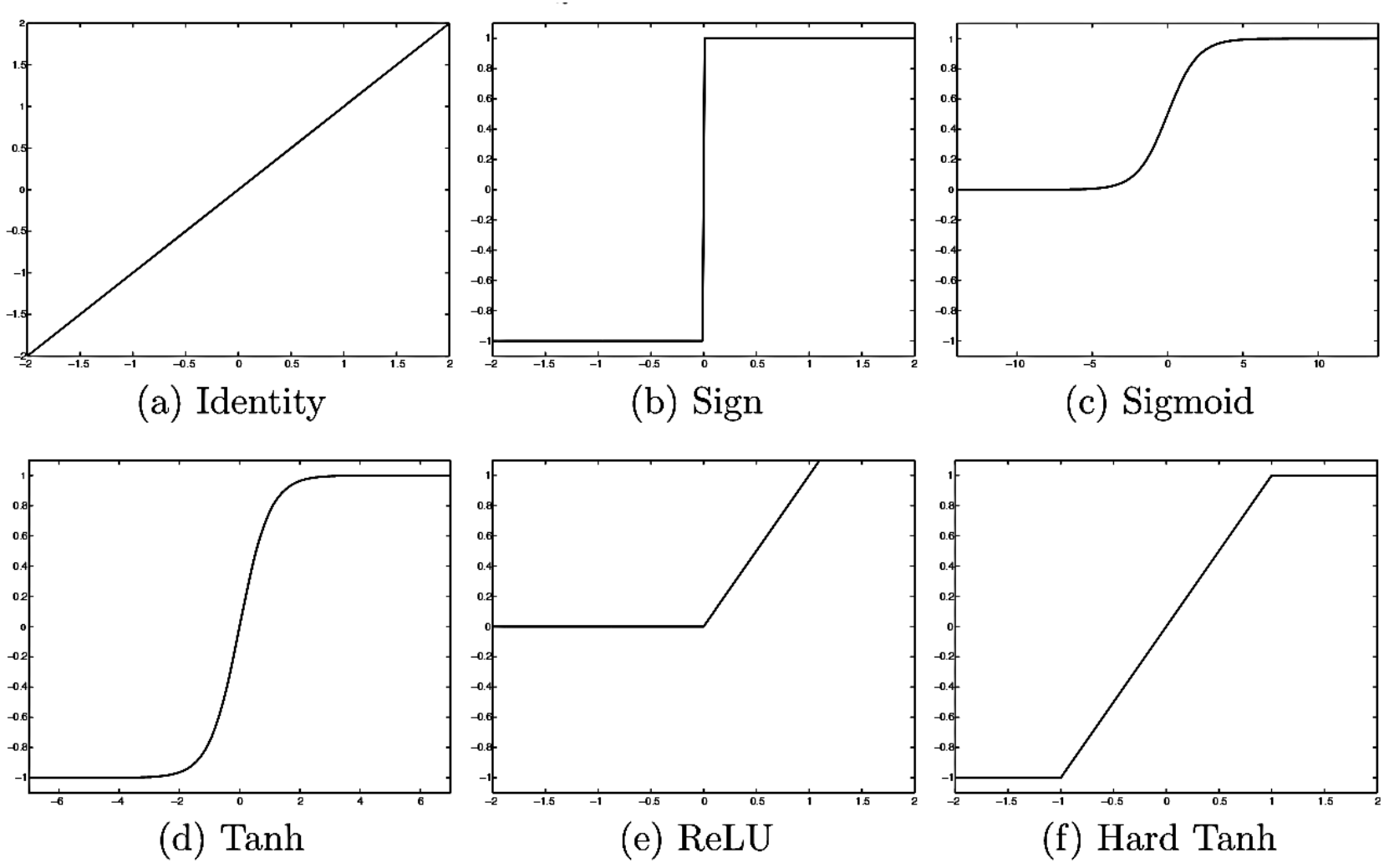


Image : C. C. Aggarwal (2018). *Neural networks and Deep Learning*, Springer.

Remarque : lien avec la régression logistique

Remarque. Avec la fonction d'activation logistique (sigmoïde),

$$y = \varphi(v) = \frac{1}{1 + e^{-v}} \quad \Leftrightarrow \quad v = \ln\left(\frac{y}{1-y}\right).$$

Puisque $v = wx + b$, on retrouve pour $h(x)$ la forme du prédicteur utilisé en régression logistique.

Plan du cours

Le perceptron multi-couches : définition

Soient p, K des entiers non nuls.

Définition : perceptron multi-couches[†] (MLP)

On appelle perceptron multi-couche à $M+1$ couches, p variables (entrées) et K réponses (sorties), une fonction $\mathbb{R}^p \to \mathbb{R}^K$ de la forme

$$h = \left(\underline{\varphi}_M \circ g_M\right) \circ \cdots \circ \left(\underline{\varphi}_j \circ g_j\right) \circ \cdots \circ \left(\underline{\varphi}_1 \circ g_1\right),$$

où[‡]

- $g_k : \mathbb{R}^{m_{k-1}} \to \mathbb{R}^{m_k}$ est affine,
- $\underline{\varphi}_k : \mathbb{R}^{m_k} \to \mathbb{R}^{m_k}$ représente l'action coordonnée par coordonnée d'une fonction $\varphi_k : \mathbb{R} \to \mathbb{R}$ croissante.
- $m_0, m_1, \ldots, m_M$: entiers non nuls, $m_0 = p$, $m_M = K$.

[†] Le perceptron original de Rosenblatt (1957) ne comportait pas de couche cachée ($M = 1$). Il utilisait la fonction d'activation $h(x) = \mathrm{sgn}(x)$ comme McCulloch et Pitts (1943), et des poids $w_j \in \{-1, +1, -\infty\}$.

[‡] à une exception près que l'on verra plus loin (couche « softmax »)

Le perceptron multi-couches : définition (suite)

Vocabulaire : couches de variables

- $z_{[0]} = x$: couche d'entrée,
- $z_{[k]} = \big(\underline{\varphi}_k \circ g_k\big)\big(z_{[k-1]}\big)$, $1 \leq k < M$: couches cachées,
- $z_{[M]} = y = \big(\underline{\varphi}_M \circ g_M\big)\big(z_{[M-1]}\big)$: couche de sortie.

Remarque. Notons

$$g_k\big(z_{[k-1]}\big) = W_k z_{[k-1]} + b_k.$$

Alors, pour tout $j \in \{1, \ldots, m_k\}$ on reconnaît un neurone :

$$z_{[k]}^{(j)} = \varphi_k\left(w_{k,j}\, z_{[k-1]} + b_k^{(j)}\right),$$

où $w_{k,j} = e_j^\top W_k$ est la j-ème ligne de W_k.

➠ Vocabulaire : poids, biais, fonction d'activation.

Le perceptron multi-couches : définition (suite)

Vocabulaire : couches de variables

- $z_{[0]} = x$: couche d'entrée,
- $z_{[k]} = \left(\underline{\varphi}_k \circ g_k\right)\left(z_{[k-1]}\right)$, $1 \leq k < M$: couches cachées,
- $z_{[M]} = y = \left(\underline{\varphi}_M \circ g_M\right)\left(z_{[M-1]}\right)$: couche de sortie.

Remarque. Notons

$$g_k\left(z_{[k-1]}\right) = W_k z_{[k-1]} + b_k.$$

Alors, pour tout $j \in \{1, \ldots, m_k\}$ on reconnaît un neurone :

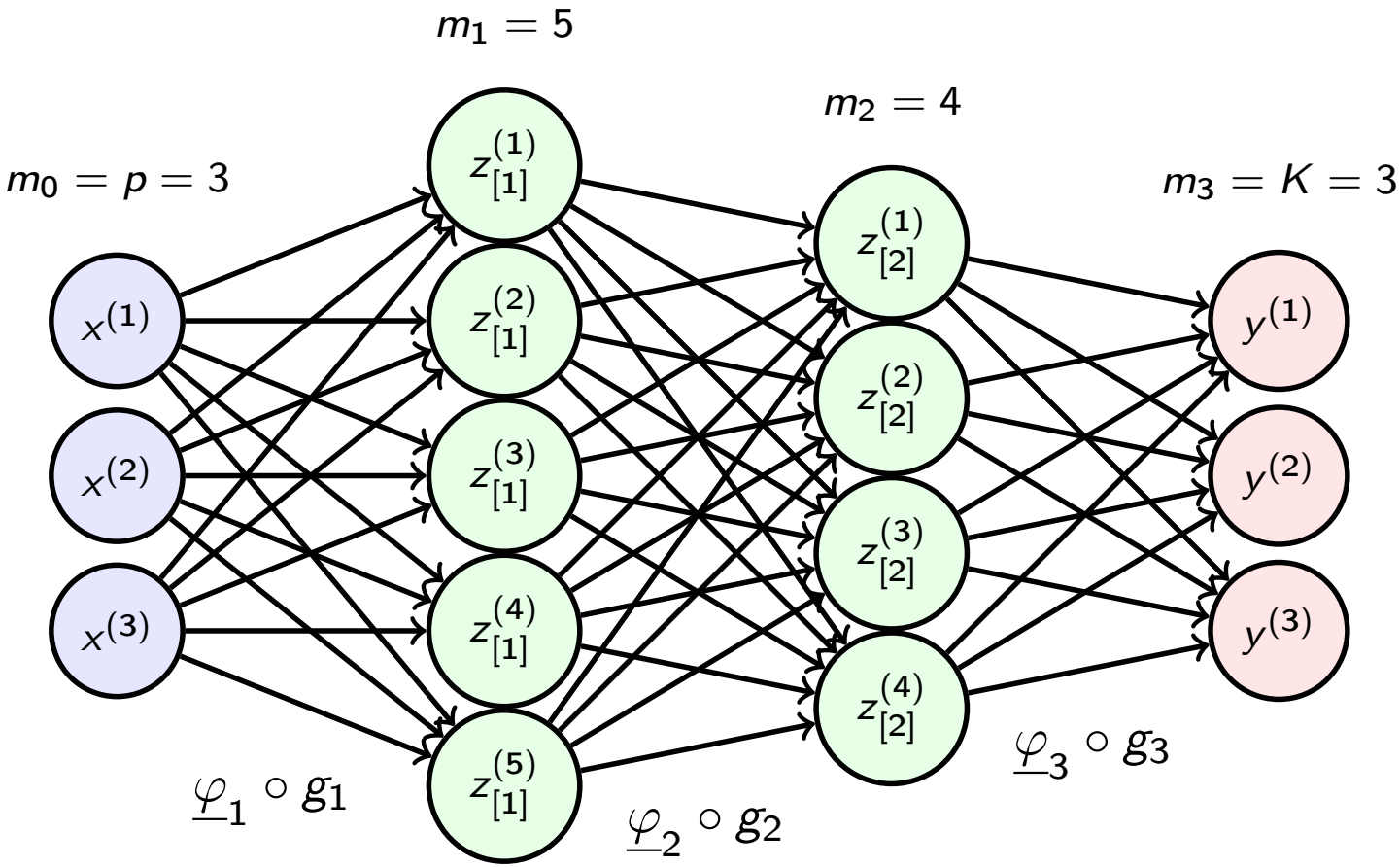
$$z_{[k]}^{(j)} = \varphi_k\left(w_{k,j}\, z_{[k-1]} + b_k^{(j)}\right),$$

où $w_{k,j} = e_j^\top W_k$ est la j-ème ligne de W_k.

➠ Vocabulaire : poids, biais, fonction d'activation.

Le perceptron multi-couches : représentation

Exemple de perceptron multi-couches avec $p = 3$ entrées, $K = 3$ sorties, et deux couches cachées de tailles $m_1 = 5$ et $m_2 = 4$.



Vocabulaire : réseau complètement connecté, à propagation avant (*fully connected, feed-forward neural network*)

Couche de sortie : fonction d'activation

La couche de sortie doit être adaptée au problème considéré...

Régression. $\mathcal{Y} \subset \mathbb{R}$, ou plus généralement $\mathbb{R}^K$.

- Perceptron à K sorties.
- Fonction d'activation : $\varphi_M = \mathrm{Id}$.
- La dernière transformation $(\varphi_M \circ g_M)$ est donc linéaire (affine).

Classification. K classes, $\mathcal{Y} = [0,1]^K$ (classification « douce »).

- Perceptron à K sorties, avec $m_{M-1} = m_M = K$.
- Exception à la définition ➠ la couche « softmax » :

$$z_{[M]}^{(j)} = \frac{\exp\left(z_{[M-1]}^{(j)}\right)}{\sum_{j'=1}^{p} \exp\left(z_{[M-1]}^{(j')}\right)}, \qquad \sum_{j=1}^{K} z_{[M]}^{(j)} = 1.$$

Remarque : pour la classification *binaire* on peut se contenter d'une seule sortie ($K = 1$ au lieu de $K = 2$) en plaçant sur la dernière couche une fonction d'activation *logistique*.

Couche de sortie : fonction d'activation

La couche de sortie doit être adaptée au problème considéré...

Régression. $\mathcal{Y} \subset \mathbb{R}$, ou plus généralement $\mathbb{R}^K$.

- Perceptron à K sorties.
- Fonction d'activation : $\varphi_M = \mathrm{Id}$.
- La dernière transformation $(\varphi_M \circ g_M)$ est donc linéaire (affine).

Classification. K classes, $\mathcal{Y} = [0,1]^K$ (classification « douce »).

- Perceptron à K sorties, avec $m_{M-1} = m_M = K$.
- Exception à la définition ➠ la couche « softmax » :

$$z_{[M]}^{(j)} = \frac{\exp\left(z_{[M-1]}^{(j)}\right)}{\sum_{j'=1}^{p} \exp\left(z_{[M-1]}^{(j')}\right)}, \qquad \sum_{j=1}^{K} z_{[M]}^{(j)} = 1.$$

Remarque : pour la classification *binaire* on peut se contenter d'une seule sortie ($K = 1$ au lieu de $K = 2$) en plaçant sur la dernière couche une fonction d'activation *logistique*.

Couche de sortie : fonction d'activation

La couche de sortie doit être adaptée au problème considéré...

Régression. $\mathcal{Y} \subset \mathbb{R}$, ou plus généralement $\mathbb{R}^K$.

- Perceptron à K sorties.
- Fonction d'activation : $\varphi_M = \mathrm{Id}$.
- La dernière transformation $(\varphi_M \circ g_M)$ est donc linéaire (affine).

Classification. K classes, $\mathcal{Y} = [0,1]^K$ (classification « douce »).

- Perceptron à K sorties, avec $m_{M-1} = m_M = K$.
- Exception à la définition ➠ la couche « softmax » :

$$z_{[M]}^{(j)} = \frac{\exp\left(z_{[M-1]}^{(j)}\right)}{\sum_{j'=1}^{p} \exp\left(z_{[M-1]}^{(j')}\right)}, \qquad \sum_{j=1}^{K} z_{[M]}^{(j)} = 1.$$

Remarque : pour la classification *binaire* on peut se contenter d'une seule sortie ($K = 1$ au lieu de $K = 2$) en plaçant sur la dernière couche une fonction d'activation *logistique*.

Apprentissage : fonctions de perte et régularisation

Les fonctions de pertes les plus couramment utilisées[†] sont

- régression : la perte quadratique
  - $L(y, \tilde{y}) = (y - \tilde{y})^2$ si une seule sortie,
  - $L(y, \tilde{y}) = \|y - \tilde{y}\|^2$ si $K > 1$.
- classification (douce) : la perte logarithmique
  - Pour tout $j \in \{1, \dots, K\}$, on a $y^{(j)} \in \{0, 1\}$ et $\tilde{y}^{(j)} \in [0, 1]$.
  - $L(y, \tilde{y}) = -\sum_{j=1}^{K} y^{(j)} \ln\left(\tilde{y}^{(j)}\right)$.

Nb paramètres élevé $\Rightarrow$ régulariser pour éviter le sur-apprentissage

- pénalisation, par exemple L^1 (LASSO) ou L^2 (ridge) ;
- autres (hors programme) : early stopping, drop out...

[†] **par exemple** `https://scikit-learn.org/stable/modules/neural_networks_supervised.html`

Apprentissage : fonctions de perte et régularisation

Les fonctions de pertes les plus couramment utilisées[†] sont

- régression : la perte quadratique
  - $L(y, \tilde{y}) = (y - \tilde{y})^2$ si une seule sortie,
  - $L(y, \tilde{y}) = \|y - \tilde{y}\|^2$ si $K > 1$.
- classification (douce) : la perte logarithmique
  - Pour tout $j \in \{1, \dots, K\}$, on a $y^{(j)} \in \{0, 1\}$ et $\tilde{y}^{(j)} \in [0, 1]$.
  - $L(y, \tilde{y}) = -\sum_{j=1}^{K} y^{(j)} \ln\left(\tilde{y}^{(j)}\right)$.

Nb paramètres élevé $\Rightarrow$ régulariser pour éviter le sur-apprentissage

- pénalisation, par exemple L^1 (LASSO) ou L^2 (ridge) ;
- autres (hors programme) : early stopping, drop out...

[†] **par exemple** `https://scikit-learn.org/stable/modules/neural_networks_supervised.html`

Apprentissage : fonctions de perte et régularisation

Les fonctions de pertes les plus couramment utilisées[†] sont

- régression : la perte quadratique
  - $L(y, \tilde{y}) = (y - \tilde{y})^2$ si une seule sortie,
  - $L(y, \tilde{y}) = \|y - \tilde{y}\|^2$ si $K > 1$.
- classification (douce) : la perte logarithmique
  - Pour tout $j \in \{1, \ldots, K\}$, on a $y^{(j)} \in \{0, 1\}$ et $\tilde{y}^{(j)} \in [0, 1]$.
  - $L(y, \tilde{y}) = -\sum_{j=1}^{K} y^{(j)} \ln\left(\tilde{y}^{(j)}\right)$.

Nb paramètres élevé $\Rightarrow$ régulariser pour éviter le sur-apprentissage

- pénalisation, par exemple L^1 (LASSO) ou L^2 (ridge) ;
- autres (hors programme) : early stopping, drop out...

[†] **par exemple** `https://scikit-learn.org/stable/modules/neural_networks_supervised.html`

Apprentissage : optimisation numérique

On doit minimiser le risque empirique (éventuellement pénalisé)

$$\hat{\mathscr{R}}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} L\left(Y_i, h_\theta(X_i)\right),$$

où θ désigne le vecteur des paramètres du modèle (poids, biais).

➠ Recours à des méthodes numériques.

Ces méthodes utilisent le gradient du critère. Deux remarques :

- alléger les calculs qd n est grand : « mini-batchs » aléatoires
  ➠ méthode du gradient stochastique (hors programme) ;
- calcul récursif du gradient d'une fonction composée
  ➠ méthode de la rétro-propagation (hors programme).

Apprentissage : optimisation numérique

On doit minimiser le risque empirique (éventuellement pénalisé)

$$\hat{\mathscr{R}}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} L\left(Y_i, h_\theta(X_i)\right),$$

où θ désigne le vecteur des paramètres du modèle (poids, biais).

➠ Recours à des méthodes numériques.

Ces méthodes utilisent le gradient du critère. Deux remarques :

- alléger les calculs qd n est grand : « mini-batchs » aléatoires
  ➠ méthode du gradient stochastique (hors programme) ;
- calcul récursif du gradient d'une fonction composée
  ➠ méthode de la rétro-propagation (hors programme).

Apprentissage : optimisation numérique

On doit minimiser le risque empirique (éventuellement pénalisé)

$$\hat{\mathscr{R}}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} L\left(Y_i, h_\theta(X_i)\right),$$

où θ désigne le vecteur des paramètres du modèle (poids, biais).

➠ Recours à des méthodes numériques.

Ces méthodes utilisent le gradient du critère. Deux remarques :

- alléger les calculs qd n est grand : « mini-batchs » aléatoires
  ➠ méthode du gradient stochastique (hors programme) ;
- calcul récursif du gradient d'une fonction composée
  ➠ méthode de la rétro-propagation (hors programme).

Plan du cours

Exemple : MNIST

70 000 images[†] de 28 × 28 pixels (256 niveau de gris)

Problème : classification multi-classes (10 classes) ;

apprentissage : 60 000 images / test : 10 000 images

Source : `http://yann.lecun.com/exdb/mnist/`

Exemple : MNIST

➠ voir notebook Jupyter / Python / Scikit-Learn

Plan du cours

Réseaux de neurones convolutionnels (CNNs)

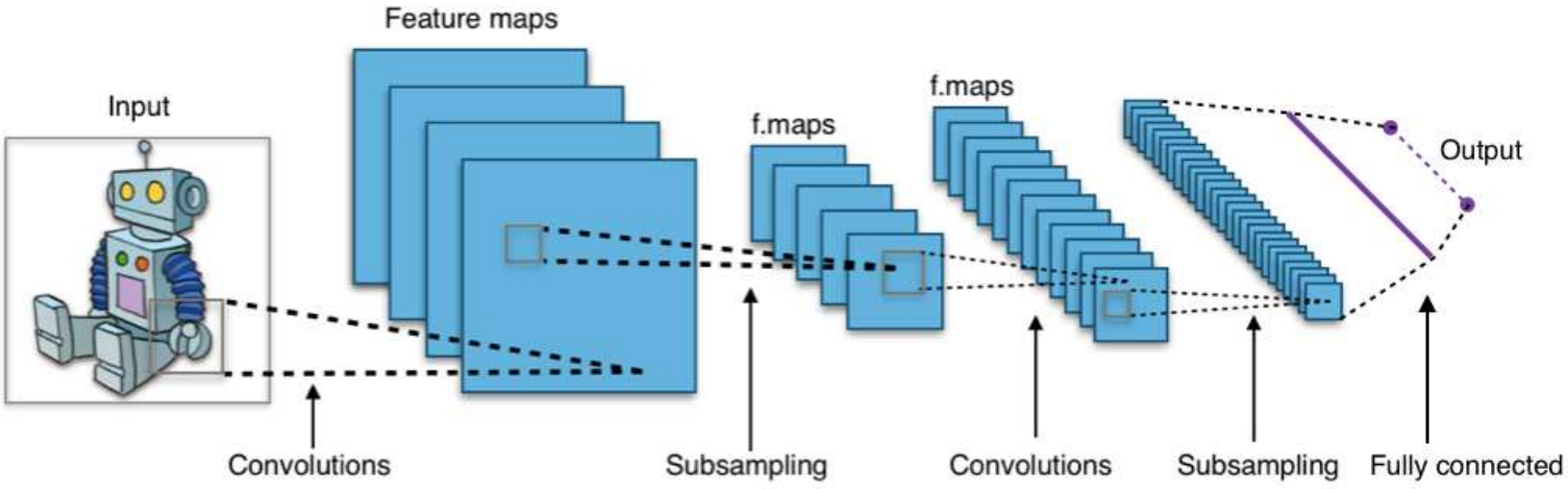


Schéma de principe d'un réseau convolutionnel typique

Image : Aphex34, `https://commons.wikimedia.org`, CC BY-SA 4.0

Plan du cours

Plan du cours

Exercice 1 (Classifieur optimal pour la perte logarithmique)

corrigé

Question

Montrer la proposition du slide 12 :

La fonction de classification $h : \mathcal{X} \to [0, 1]$ est optimale pour la perte logarithmique ssi, P^X-pp,

$$h(x) \;=\; \mathrm{P}\left(Y = 1 \mid X = x\right).$$

Exercice 2 (Régression logistique multi-classes)

corrigé

Lorsque le nombre de classes K est supérieur ou égal à 3, on parle de Classification multi-classes.

Soit $\{0, 1, \ldots, K-1\}$ l'ensemble des labels (classes), $K \geq 3$.

La régression logistique binaire (2 classes) s'étend au cas multi-classes en :

- considérant une classe (ici « 0 ») comme référence,
- réalisant $K-1$ régressions logistiques binaires :

$$\begin{cases} \ln\left(\frac{\mathrm{P}(Y=1|X=x)}{\mathrm{P}(Y=0|X=x)}\right) & = & \beta_{1,0} + \beta_1^\top x \\ \vdots & & \\ \ln\left(\frac{\mathrm{P}(Y=K-1|X=x)}{\mathrm{P}(Y=0|X=x)}\right) & = & \beta_{K-1,0} + \beta_{K-1}^\top x \end{cases}$$

Exercice 2 (Régression logistique multi-classes (suite)) corrigé

Afin d'alléger les notations, on supposera que la matrice des variables explicatives contient un vecteur constant et on fera ainsi le changement : $\beta_k \leftarrow (\beta_{k,0}, \beta_k)$

Questions

1. Exprimer $\mathrm{P}(Y = k|X = x)$ en fonction des β_k,
2. En déduire que le choix de la classe de référence n'a pas d'influence sur le modèle de régression,
3. Ecrire la log-vraisemblance

Plan du cours

Corrigé de l'exercice 1

retour au sujet

Le classifieur optimal $h^\star$ minimise $\mathbb{E}\left(L(Y, h(X))\right)$, l'espérance portant sur (X, Y).

Par conditionnement, il vient :

$$\mathbb{E}_{(X,Y)}\left(L(Y, h(X))\right) = \mathbb{E}_X\left(\mathbb{E}_{Y|X}(L(Y, h(X))|X\right)$$

Ainsi :

$$h^\star = \operatorname{argmin}_h \ \mathbb{E}\left(L(Y, h(X))\right)$$

$$\Updownarrow$$

$$h^\star(x) = \operatorname{argmin}_{t\in\mathcal{Y}} \underbrace{\mathbb{E}\left(L(Y, t) \mid X = x\right)}_{\mathcal{J}(t)} \quad \mathrm{P}^X\text{-pp.}$$

Corrigé de l'exercice 1

retour au sujet

Avec la perte logarithmique :

$$\begin{aligned}
\mathcal{I}(t) &= \mathbb{E}_{Y|X}\left(L(Y,t)|X=x\right) \\
&= \mathbb{P}(Y=1|X=x)L(1,t) + \mathbb{P}(Y=0|X=x)L(0,t) \\
&= \mathbb{P}(Y=1|X=x)(L(1,t)-L(0,t)) + L(0,t) \\
&= \mathbb{P}(Y=1|X=x)(-\ln(t)+\ln(1-t)) - \ln(1-t)
\end{aligned}$$

Corrigé de l'exercice 1

retour au sujet

La minimisation de $\mathcal{J}(t)$ s'obtient en étudiant le signe de la dérivée de $\mathcal{J}(t)$:

$$\begin{aligned}\mathcal{J}'(t) &= \mathbb{P}(Y=1|X=x)\left(-\frac{1}{t}-\frac{1}{1-t}\right)+\frac{1}{1-t}\\ &= \mathbb{P}(Y=1|X=x)\left(-\frac{1}{t(1-t)}\right)+\frac{1}{1-t}\\ &= \frac{1}{1-t}\left(1-\frac{\mathbb{P}(Y=1|X=x)}{t}\right)\end{aligned}$$

Pour $\mathbb{P}(Y=1|X=x)\in]0,1[$, $\mathcal{J}'(t)$ est donc :

- strictement négative pour $t\in]0,\mathbb{P}(Y=1|X=x)[$,
- nulle en $t=\mathbb{P}(Y=1|X=x)$,
- strictement positive pour $t\in]\mathbb{P}(Y=1|X=x),1[$,

Corrigé de l'exercice 1

retour au sujet

Bilan : $t = \mathbb{P}(Y = 1|X = x)$ est l'unique minimiseur sur $[0, 1]$ de la fonction $\mathcal{J}(t)$.

Lorsque

- $\mathbb{P}(Y = 1|X = x) = 0$, $\mathcal{J}(t)$ est minimale en $t = 0$,
- $\mathbb{P}(Y = 1|X = x) = 1$, $\mathcal{J}(t)$ est minimale en $t = 1$.

On en conclut que :

1. $h^\star : x \mapsto \mathbb{P}(Y = 1|X = x)$ est optimale,
2. $h^\star(x)$ est l'unique minimiseur de la fonction $\mathcal{J} : t \mapsto \mathbb{E}_{Y|X}\left(L(Y, t)|X = x\right)$.

Réciproquement : supposons h optimale.
Soit la fonction :

$$g(x) = \mathbb{E}_{Y/X}(L, Y, h(X))/X = x) - \mathbb{E}_{Y/X}(L, Y, h^\star(X))/X = x)$$

On a :

- $\mathbb{E}_X(g(X)) = \int_{\mathcal{X}} g(x) d\mathrm{P}^X = 0$ (sinon h ne serait pas optimale),
- $\forall x,\ g(x) \geq 0$ par optimalité de $h^\star$.

Ainsi $g = 0\ \mathrm{P}^X -$ pp, c'est-à-dire :

$$\mathbb{P}(X \in F) = 0 \text{ avec } F = \{x \in \mathcal{X} \text{ t.q. } g(x) > 0\}$$

Corrigé de l'exercice 1

retour au sujet

$h^\star(x)$ étant l'unique minimiseur de la fonction $\mathcal{J} : t \mapsto \mathbb{E}_{Y|X}\left(L(Y,t)|X=x\right)$, on a :

$$F = \{x \in \mathcal{X} \text{ t.q. } h(x) \neq h^\star(x)\}$$

On en conclut que $h = h^\star$ $\mathrm{P}^X - p.p.$, d'où l'équivalence cherchée.

Corrigé de l'exercice 2

retour au sujet

❶ Pour $k \in \{1, \ldots, K-1\}$:

$$\mathrm{P}(Y = k|X = x) = \exp\left(\beta_k^\top x\right) \mathrm{P}(Y = 0|X = x).$$

Utilisant $\sum_{k=0}^{K-1} \mathrm{P}(Y = k|X = x) = 1$, il vient :

$$\begin{cases} \mathrm{P}(Y = k|X = x) &= \dfrac{\exp\left(\beta_k^\top x\right)}{1 + \sum_{k' \neq 0} \exp\left(\beta_{k'}^\top x\right)}, \ k \neq 0 \\ \mathrm{P}(Y = 0|X = x) &= \dfrac{1}{1 + \sum_{k' \neq 0} \exp\left(\beta_{k'}^\top x\right)} \end{cases} \tag{1}$$

retour au sujet

❷ Soit $\tilde{k} \neq 0$. Supposons que l'on ait pris la classe $\tilde{k}$ comme référence et soit pour $k \neq \tilde{k}$ le vecteur $\tilde{\beta}_k$ du modèle :

$$\ln\left(\frac{\mathrm{P}(Y=k|X=x)}{\mathrm{P}\left(Y=\tilde{k}|X=x\right)}\right) = \tilde{\beta}_k^\top x,\ k \neq \tilde{k}$$

Comme

$$\begin{aligned}\ln\left(\frac{\mathrm{P}(Y=k|X=x)}{\mathrm{P}\left(Y=\tilde{k}|X=x\right)}\right) &= \ln\left(\frac{\mathrm{P}(Y=k|X=x)}{\mathrm{P}(Y=0|X=x)}\frac{\mathrm{P}(Y=0|X=x)}{\mathrm{P}\left(Y=\tilde{k}|X=x\right)}\right)\\ &= \beta_k^\top x - \beta_{\tilde{k}}^\top x,\end{aligned}$$

il vient :

$$\left\{\begin{array}{rcl}\tilde{\beta}_k &=& \beta_k - \beta_{\tilde{k}},\ k \neq 0,\ k \neq \tilde{k}\\ \tilde{\beta}_0 &=& -\beta_{\tilde{k}}\end{array}\right. \qquad (2)$$

Corrigé de l'exercice 2

retour au sujet

Utilisant le résultat de la question ❶, on a, pour $k \neq \tilde{k}$:

$$\begin{cases} \mathrm{P}(Y = k|X = x) &= \dfrac{\exp\left(\tilde{\beta}_k^\top x\right)}{1 + \sum_{k' \neq \tilde{k}} \exp\left(\tilde{\beta}_{k'}^\top x\right)}, \ k \neq \tilde{k} \\ \mathrm{P}(Y = 0|X = x) &= \dfrac{1}{1 + \sum_{k' \neq \tilde{k}} \exp\left(\tilde{\beta}_{k'}^\top x\right)} \end{cases}$$

En substituant dans ces équations les $\tilde{\beta}_k$ à l'aide des relations (2), il vient les équations (1).

Bilan. Changer de référence revient à paramétrer autrement le modèle (sans changer de modèle).

Corrigé de l'exercice 2

retour au sujet

❸ Afin de faire le lien avec la régression logistique binaire, on représente l'obsevation $y_i \in \{0, \dots, K-1\}$ par le vecteur $z_i \in \{0,1\}^K$:

$$z_{i,k} = \begin{cases} 1 & \text{si } y_i = k \\ 0 & \text{sinon} \end{cases}$$

On note également $\boldsymbol{\beta}$ l'ensemble des vecteurs $\beta_1, \dots, \beta_{K-1}$.

Corrigé de l'exercice 2

retour au sujet

On écrit la vraisemblance associée à l'exemple (x_i, y_i) :

$$\begin{aligned}
\mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(y_i|x_i) &= \prod_{k=0}^{K-1} \mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(k|x_i)^{z_{i,k}} \\
&= \mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(0|x_i)^{1-z_{i,1}-\ldots z_{i,K-1}} \left(\prod_{k=1}^{K-1} \mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(k|x_i)^{z_{i,k}} \right) \\
&= \mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(0|x_i) \prod_{k=1}^{K-1} \left(\frac{\mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(k|x_i)}{\mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(0|x_i)} \right)^{z_{i,k}}
\end{aligned}$$

Corrigé de l'exercice 2

retour au sujet

Utilisant :

- les $(K-1)$ modèles de régression :

$$\frac{\mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(k|x_i)}{\mathrm{P}_{\boldsymbol{\beta}}^{Y_i|X_i}(0|x_i)} = \exp(\beta_k^\top x_i)^{z_{i,k}},$$

- la deuxième équation de (1) :

$$\mathrm{P}(Y=0|X=x) = \frac{1}{1+\sum_{k'=1}^{K-1}\exp\left(\beta_{k'}^\top x\right)},$$

il vient la log-vraisemblance :

$$\ell(\boldsymbol{\beta}) = \sum_{k=1}^{K-1}\sum_{i=1}^{n} z_{i,k}\beta_k^\top x_i - \ln\left(1+\sum_{k'=1}^{K-1}\exp\left(\beta_{k'}^\top x_i\right)\right)$$

Plan du cours

Modèle linéaire généralisé

GLM = Generalized Linear Model

Il regroupe les modèles pour lesquels :

- $Y|X$ suit une loi issue d'une famille exponentielle :

$$f^{Y|X}(y|x) = C(\eta)h(y)\exp(\eta y) \quad \text{avec } \eta = \eta(x).$$

- $g(\mathbb{E}_\beta(Y|X=x)) = \beta_0 + \beta^\top x$.

Vocabulaire. La fonction g s'appelle fonction de lien.[†]

Exemple. Les lois de Bernoulli forment une famille exponentielle.

$$\begin{aligned} f(y) &= \theta^y (1-\theta)^{1-y} \\ &= (1-\theta)\exp\left(\ln\left(\frac{\theta}{1-\theta}\right) y\right) \qquad \rightarrow \eta = \ln\left(\frac{\theta}{1-\theta}\right) \end{aligned}$$

[†] Si N désigne l'ensemble des valeurs admissibles du paramètre η, on choisit le plus souvent pour g une bijection de N dans $\mathbb{R}$.

Modèle linéaire généralisé

GLM = Generalized Linear Model

Il regroupe les modèles pour lesquels :

- $Y|X$ suit une loi issue d'une famille exponentielle :

$$f^{Y|X}(y|x) = C(\eta)h(y)\exp(\eta y) \quad \text{avec } \eta = \eta(x).$$

- $g(\mathbb{E}_\beta(Y|X=x)) = \beta_0 + \beta^\top x.$

Vocabulaire. La fonction g s'appelle fonction de lien.[†]

Exemple. Les lois de Bernoulli forment une famille exponentielle.

$$\begin{aligned} f(y) &= \theta^y (1-\theta)^{1-y} \\ &= (1-\theta)\exp\left(\ln\left(\frac{\theta}{1-\theta}\right) y\right) \qquad \Rightarrow \eta = \ln\left(\frac{\theta}{1-\theta}\right) \end{aligned}$$

[†] Si N désigne l'ensemble des valeurs admissibles du paramètre η, on choisit le plus souvent pour g une bijection de N dans $\mathbb{R}$.

Remarque : modèle linéaires généralisés (GLM)

Forme du modèle considéré

- $Y|X \sim \text{Ber}\left(\mathbb{E}_\beta(Y|X)\right)$,
- $g\left(\mathbb{E}_\beta(Y|X)\right) = \beta_0 + \beta^\top X$, avec $g = \text{logit}$.

➠ cas particulier du modèle linéaire généralisé (GLM)
(g s'appelle la fonction de lien)

➠ complément

Remarque : nous avons déjà rencontré un autre exemple de GLM

- $Y|X \sim \mathcal{N}\left(\mathbb{E}_\beta(Y|X), \sigma^2\right)$
- $g\left(\mathbb{E}_\beta(Y|X)\right) = \beta_0 + \beta^\top X$ avec $g = \text{Id}$

Remarque : modèle linéaires généralisés (GLM)

Forme du modèle considéré

- $Y|X \sim \mathrm{Ber}\left(\mathbb{E}_\beta(Y|X)\right)$,
- $g\left(\mathbb{E}_\beta(Y|X)\right) = \beta_0 + \beta^\top X$, avec $g = \mathrm{logit}$.

➠ cas particulier du modèle linéaire généralisé (GLM)
(g s'appelle la fonction de lien)

➠ complément

Remarque : nous avons déjà rencontré un autre exemple de GLM

- $Y|X \sim \mathcal{N}\left(\mathbb{E}_\beta(Y|X), \sigma^2\right)$
- $g\left(\mathbb{E}_\beta(Y|X)\right) = \beta_0 + \beta^\top X$ avec $g = \mathrm{Id}$

Exemple : $Y_i|X_i \overset{\text{iid}}{\sim} \text{Poisson}(\theta_i)$, avec $\ln \theta_i = \beta_0 + \beta_1 X_i$

Les lois de Poisson forment une famille exponentielle :

$$f(y) = \exp(-\theta)\,\frac{\theta^y}{y!}$$

$$= \frac{1}{y!}\exp(-\theta)\exp(\ln(\theta)y) \qquad \Rightarrow \eta = \ln(\theta)$$