EE-311—Apprentissage et intelligence artificielle

2. Apprentissage supervisé et fonctions coût

Michael Liebling

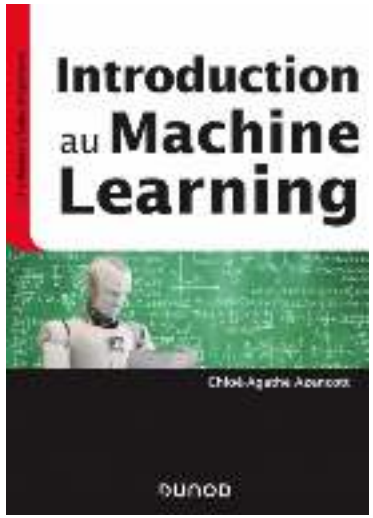
https://moodle.epfl.ch/course/view.php?id=16090

28 février 2025 (compilé le 27 février 2025)

idiap RESEARCH INSTITUTE

EPFL

Ouvrage de référence et source



Ces transparents sont basés en grande partie sur le texte de Chloé-Agathe Azencott “Introduction au Machine Learning”, Dunod, 2019
ISBN 978-210-080153-4

L’auteure a mis le texte (sans les exercices) à disposition ici : http://cazencott.info/dotclear/public/lectures/IntroML_Azencott.pdf

Avertissement : Bien que ces transparents partagent la notation mathématique, la structure de l’exposition (en partie), et certains exemples avec le livre, ils ne constituent qu’un complément et non un remplacement ou une source unique pour la couverture des matières du cours. À ce titre, ces transparents ne se substituent pas au texte.

Contenu cours 2

- Formalisation d'un problème d'apprentissage supervisé
- Fonctions de coût

Formalisation d'un problème d'apprentissage supervisé

Un problème d'apprentissage supervisé peut être formalisé de la façon suivante :

Étant données n observations $\{\vec{x}^1, \ldots, \vec{x}^n\}$, où chaque observation $\vec{x}^i$ est un élément de l'espace des observations $\mathcal{X}$, et leurs étiquettes $\{y^1, \ldots, y^n\}$, où chaque étiquette y^i appartient à l'espace des étiquettes $\mathcal{Y}$, le but de l'apprentissage supervisé est de trouver une fonction $f : \mathcal{X} \to \mathcal{Y}$ telle que $f(\vec{x}) \approx y$ pour toutes les paires $(\vec{x}, y) \in \mathcal{X} \times \mathcal{Y}$ ayant la même relation que les paires observées.

L'ensemble $\mathcal{D} = \left\{\vec{x}^i, y^i\right\}_{i=1,\ldots,n}$ forme le jeu d'apprentissage.

Différents cas d'apprentissage supervisé

Régression : $\mathcal{Y} = \mathbb{R}$
Classification binaire : $\mathcal{Y} = \{0, 1\}$ ou $\mathcal{Y} = \{-1, 1\}$
Classification multi-classe : $\mathcal{Y} = \{1, 2, \ldots, C\}$, $C > 2$
On a souvent :

$$\mathcal{X} = \mathbb{R}^p \Leftrightarrow \text{les observations sont représentées par } p \text{ variables.}$$

Dans ce cas, on on peut définir la *matrice de données* ou *matrice de design* :

$$X \in \mathbb{R}^{n \times p}$$

dont l'élément à la i-ème ligne et j-ème colonne,

$$X_{ij} = x_j^i$$

représente la j-ème variable de la i-ème observation.

Organisation des données d'un problème d'apprentissage supervisé

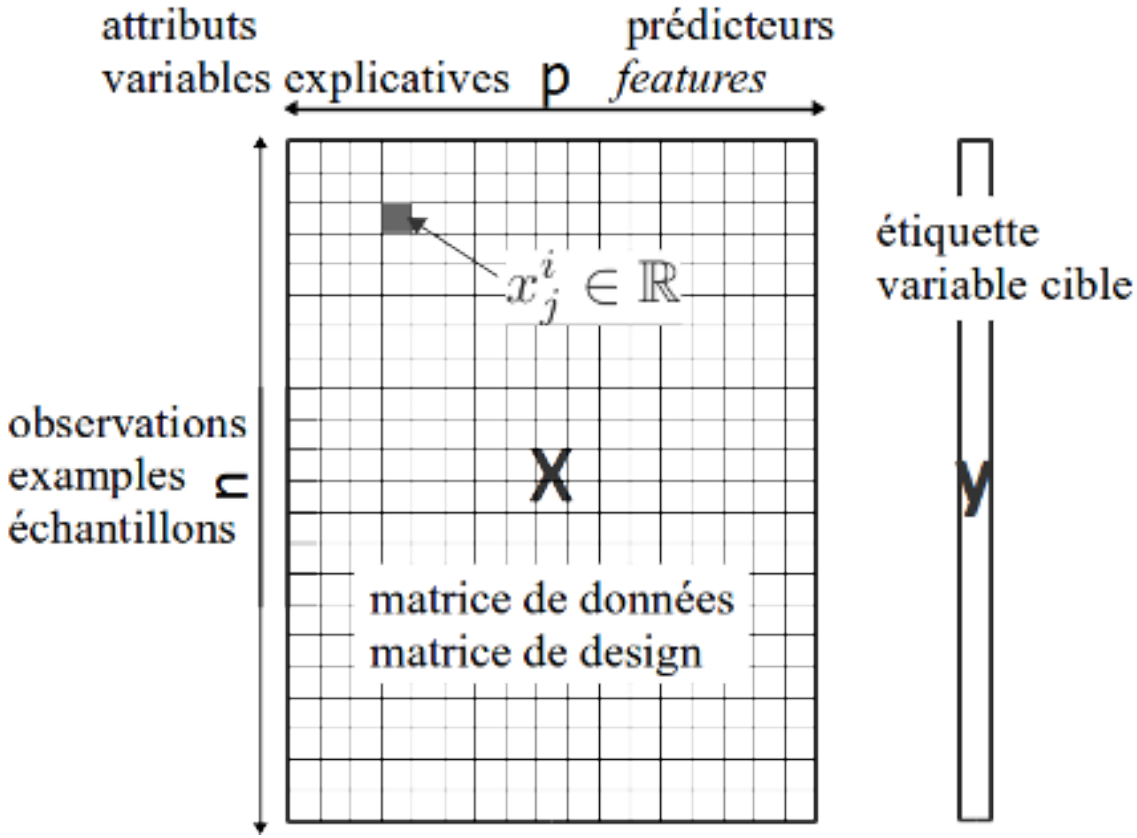


FIGURE 2.1 – Les données d'un problème d'apprentissage supervisé sont organisées en une matrice de design et un vecteur d'étiquettes. Les observations sont représentées par leurs variables explicatives.

Azencott

Classification par seuillage (cas binaire) ou maximum (multi-classe) de fonction de décision

Définition 2.1 (fonction de décision)

• **Cas binaire :** La fonction de classification $f(\vec{x})$ prend des valeurs dans $\{0, 1\}$. On appelle *fonction de décision*, ou *fonction discriminante*, une fonction intermédiaire $g : \mathcal{X} \to \mathbb{R}$ telle que :

$$f(\vec{x}) = 0 \text{ si et seulement si } g(\vec{x}) \leq 0 \text{ et}$$
$$f(\vec{x}) = 1 \text{ si et seulement si } g(\vec{x}) > 0.$$

• **Cas de la classification multi-classe** ($f(\vec{x})$ dans $\{1, \ldots, C\}$, $C > 2$) : On a C fonctions de décision $g_c : \mathcal{X} \to \mathbb{R}$, $c = 1, \ldots, C$, c.-à-d.

$$g_1 : \mathcal{X} \to \mathbb{R}$$
$$\vdots$$
$$g_C : \mathcal{X} \to \mathbb{R}$$

telles que
$$f(\vec{x}) = \arg \max_{c=1,\ldots,C} g_c(\vec{x})$$

Régions de décision

Définition 2.2 (région de décision)

• **Cas binaire :** les régions de décision $\mathscr{R}_0$ et $\mathscr{R}_1$ résultent du partitionnement de l'espace des observations $\mathscr{X}$ par la fonction discriminante g :

$$\mathscr{R}_0 = \{\vec{x} \in \mathscr{X} | g(\vec{x}) \leq 0\} \text{ et}$$
$$\mathscr{R}_1 = \{\vec{x} \in \mathscr{X} | g(\vec{x}) > 0\}$$

• **Cas multi-classes :** C régions de décision (cas une contre toutes, voir ci-après) :

$$\mathscr{R}_c = \left\{ \vec{x} \in \mathscr{X} \,\middle|\, g_c(\vec{x}) = \max_k g_k(\vec{x}) \right\}$$

Frontières de décision d'un problème de classification

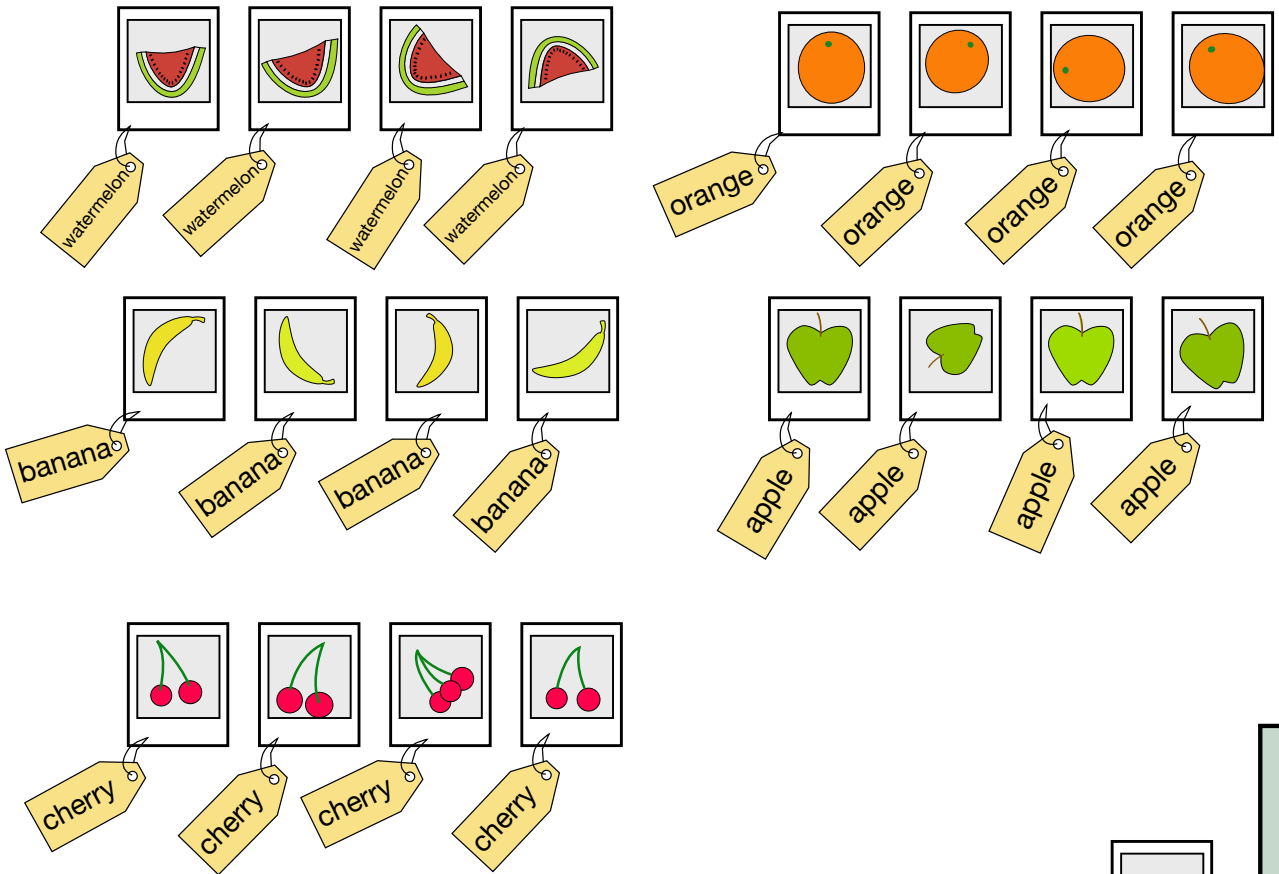
Définition 2.3 (frontière de décision ou discriminant)
l'ensemble des points de $\mathcal{X}$ où une fonction de décision s'annule.

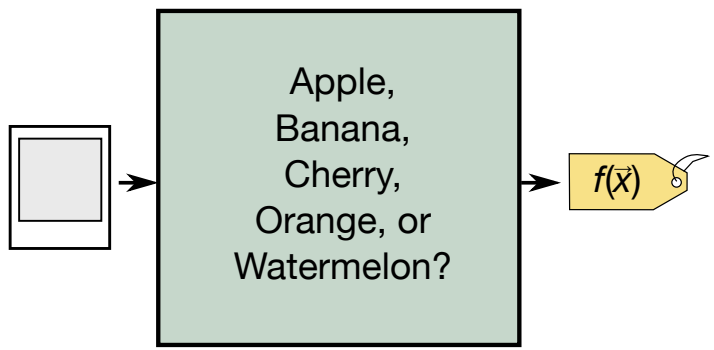
Remarques : le nombre de frontières de décision est :

- classification binaire : 1 seule frontière
- classification multi-classe à C classes : C frontières
  (chacune des $C > 2$ fonctions de décision a sa propre frontière).

Classification multi-classe : comment s'y prendre ?

Training data





Trained Classifier

Classification multi-classe avec classifieurs binaires

On peut utiliser tout algorithme de classification binaire pour résoudre un problème de classification à C classes.
Deux possibilités :

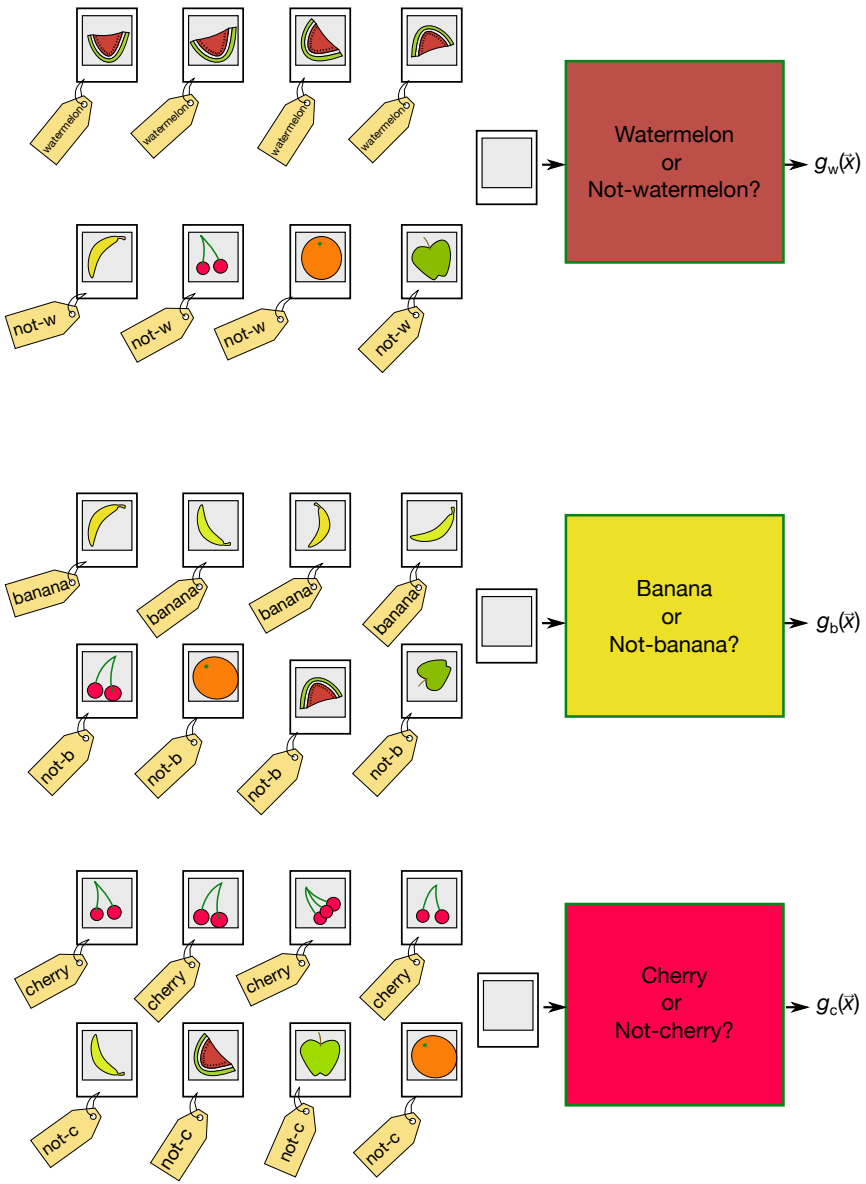
Définition 2.4 : Approche une-contre-toutes (one-versus-all)
1. Entraîner C classifieurs binaires "classe c : oui/non ?" sur l'ensemble des données d'entraînement (les exemples de la classe c sont positifs, tous les autres exemples sont négatifs)
2. Classifieur multi-classe obtenu via :

$$f(\vec{x}) = \arg \max_{c=1,\dots,C} g_c(\vec{x})$$

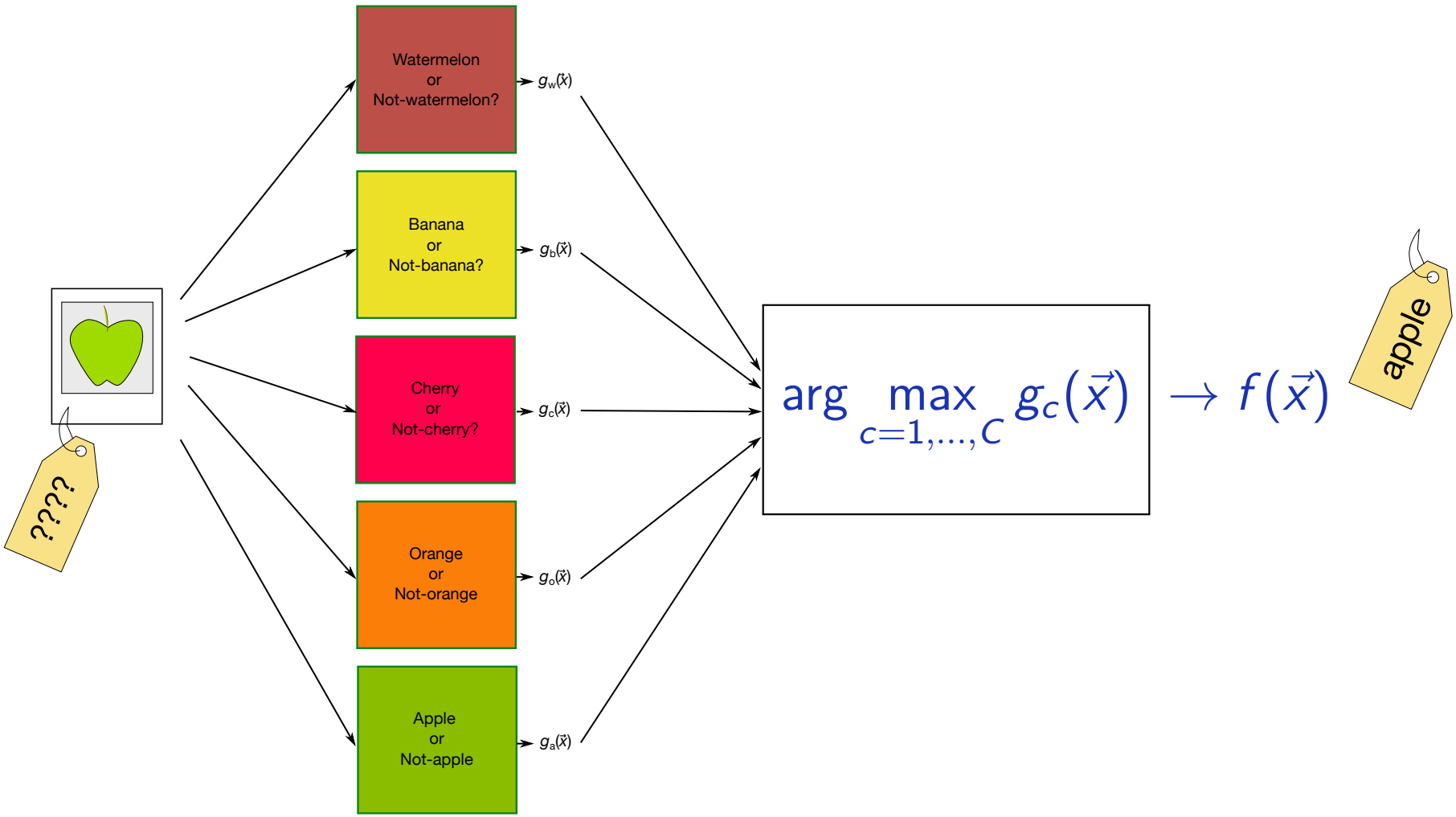
Définition 2.5 : Approche une-contre-une (one-versus-one)
1. Entraîner $C(C-1)$ classifieurs binaires "classe c : oui/non ?" sur exemples étiquetés des classes c (exemples $+$) et k (exemples -)
2. Classifieur multi-classe obtenu via :

$$f(\vec{x}) = \arg \max_{c=1,\dots,C} \left(\sum_{k \neq c} g_{ck}(\vec{x}) \right)$$

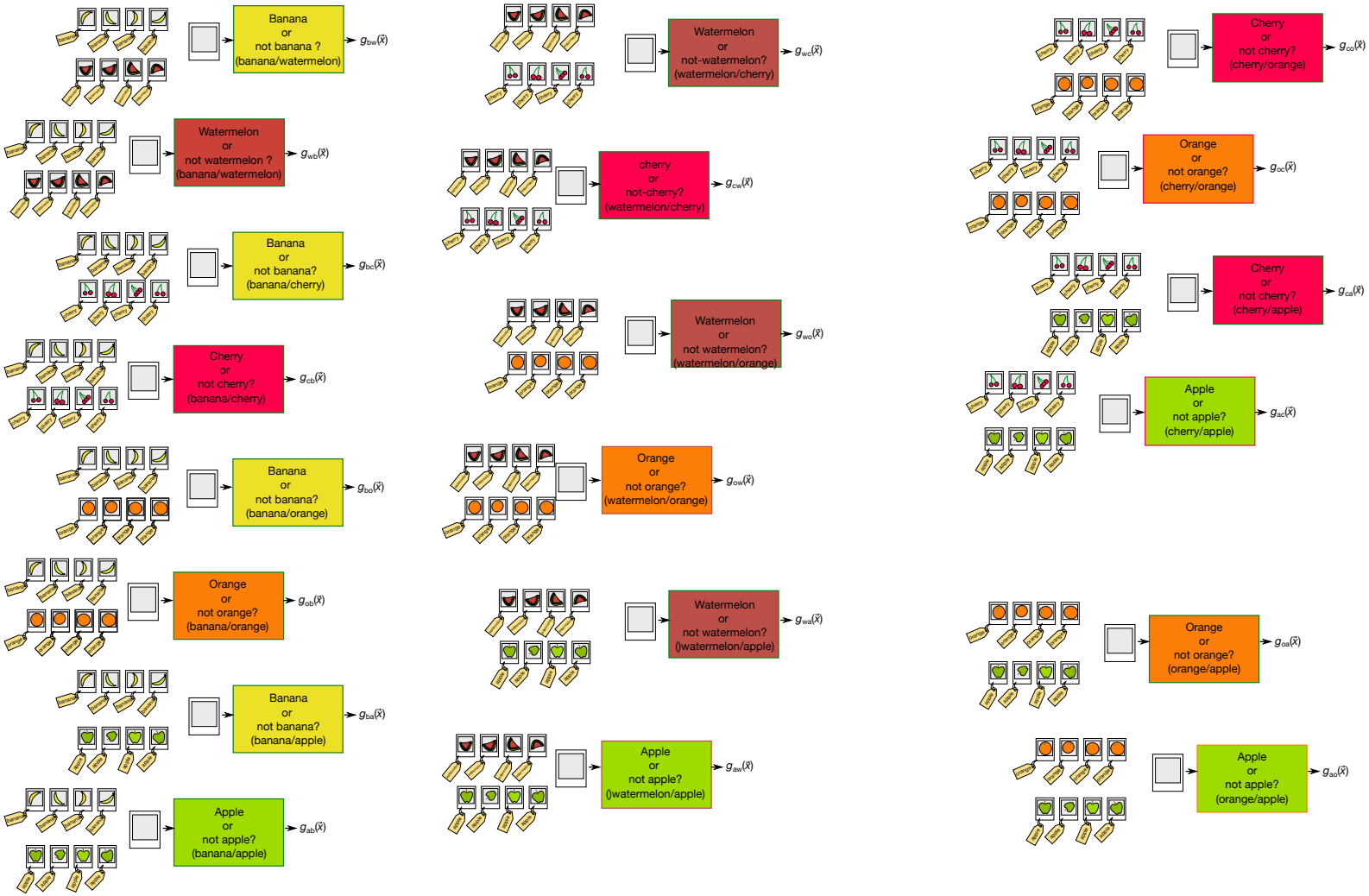
Approche une-contre-toutes : on entraîne C classifieurs binaires (exemple : $C = 5$ classifieurs)



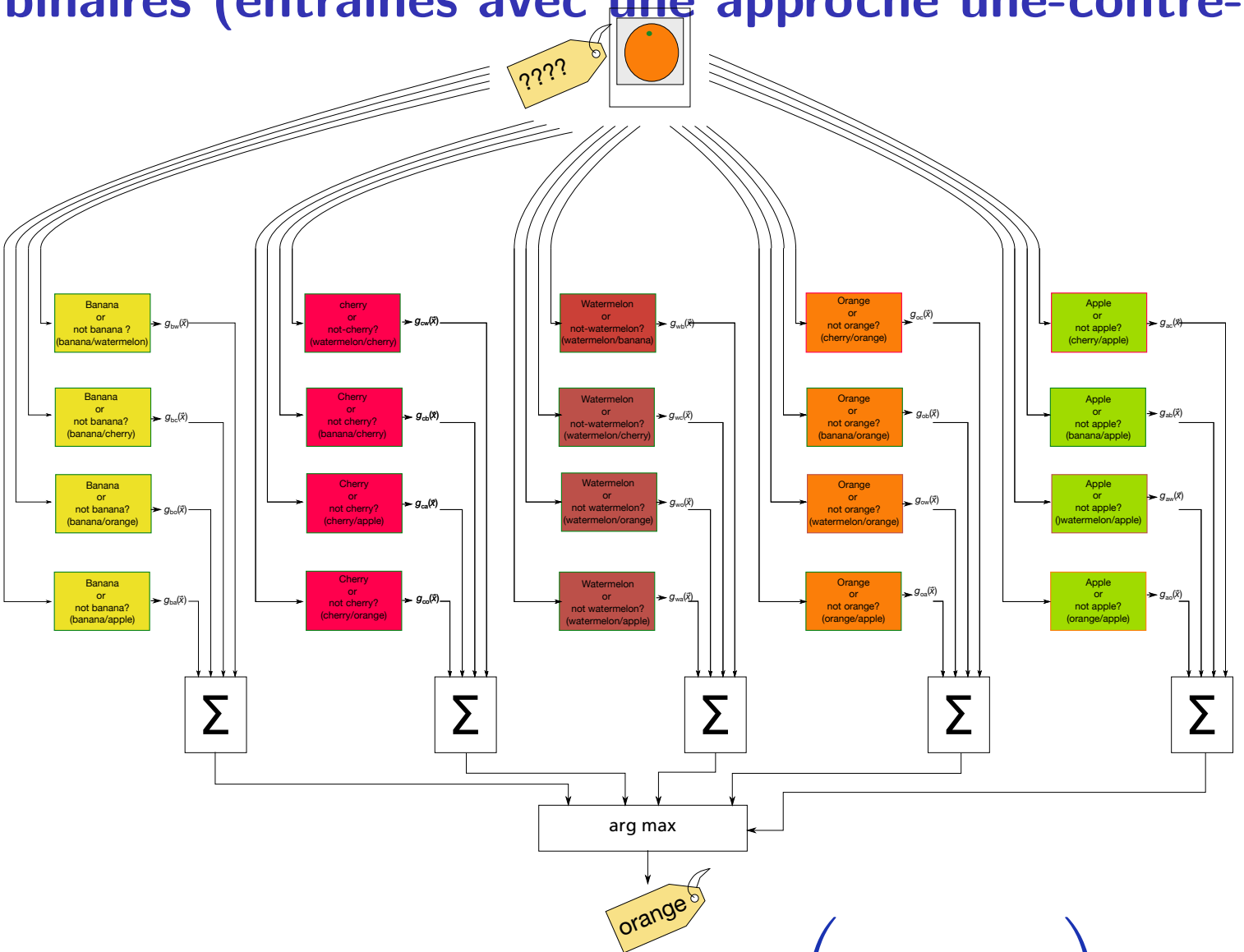
Classifieur multi-classe construit à partir de classifieurs binaires (entraînés avec une approche une-contre-toutes)



Approche une-contre-une : on entraîne $C\,(C-1)$ classifieurs binaires sur des paires de classes (exemple : $C(C-1) = 5 \times 4 = 20$ classifieurs)



Classifieur multi-classe construit à partir de classifieurs binaires (entraînés avec une approche une-contre-une)



$$f(\vec{x}) = \arg \max_{c=1,\ldots,C} \left(\sum_{k \neq c} g_{ck}(\vec{x}) \right)$$

Une-contre-toutes ou Une-contre-une ?

Suivant la taille des données n, le nombre de classes C, le coût pour entraîner un classifieur binaire et la puissance de calcul à disposition (possibilité d'entraîner plusieurs classifieurs en parallèle), on préfèrera l'une ou l'autre approche.

Efficacité de l'entraînement (on suppose que les tailles des classes d'entraînement sont égales) :

- entraîner C modèles sur n observations *ou*
- entraîner $C(C-1)$ modèles sur $2n/C$ observations ?

Qualité de l'entraînement :

- entraîner C modèles sur n observations *ou*
- entraîner $C(C-1)$ modèles sur $2n/C$ observations ?

Espace des hypothèses

Définition 2.6, espace des hypothèses : sous-espace $\mathscr{F} \subseteq \mathscr{Y}^{\mathscr{X}}$ des fonctions de modélisation de $\mathscr{X} \to \mathscr{Y}$ (aussi noté $\mathscr{Y}^{\mathscr{X}}$) qu'on considère (note : il est *choisi* pour être adapté au problème)

Exemple : espace des hypothèses qui réunit des fonctions dont les lignes de niveaux sont des ellipses, avec centre et axes ajustables selon la valeur des paramètres a, b, α, β :

$$\mathscr{F} = \left\{ \vec{x} \mapsto \alpha(x_1 - a)^2 + \beta(x_2 - b)^2 - 1 \right\}$$

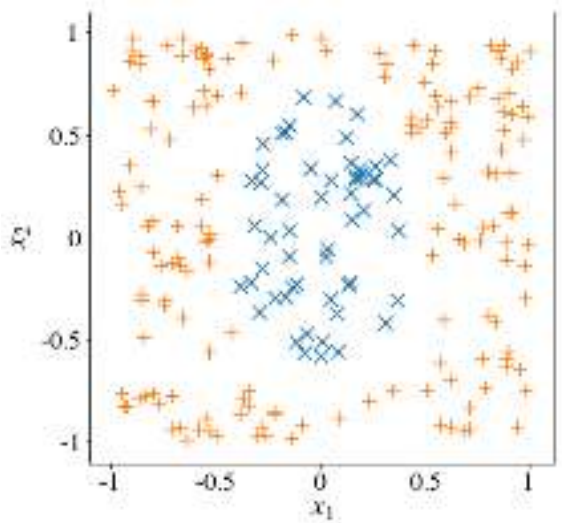
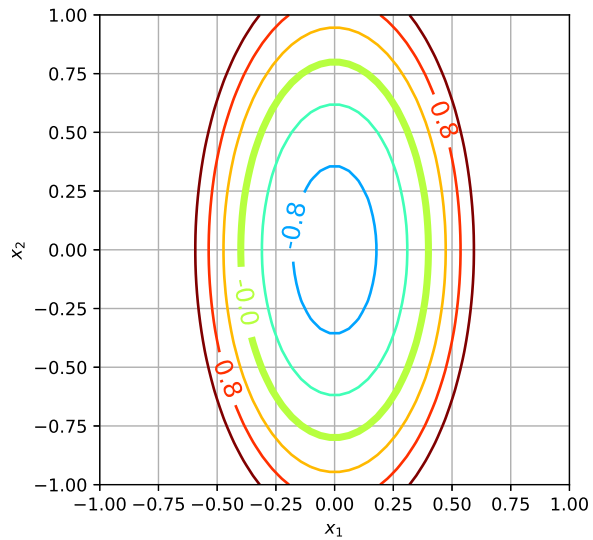


FIGURE 2.2 – Les exemples positifs (+) et négatifs (x) semblent être séparables par une ellipse. Azencott



$$\alpha = \frac{1}{0.4^2}, \ \beta = \frac{1}{0.8^2}$$

Tâche d'apprentissage supervisé

Donné :

1. jeu de n observations étiquetées : $\mathscr{D} = \left\{(\vec{x}^i, y^i)\right\}_{i=1,\ldots,n}$
2. espace d'hypothèses $\mathscr{F} \subseteq \mathscr{Y}^{\mathscr{X}}$ (un sous-espace de fonctions définies sur le domaine des données $\mathscr{X}$ vers les étiquettes $\mathscr{Y}$, $\mathscr{X} \to \mathscr{Y}$).

Supposition : les étiquettes y^i ont été calculées par une fonction $\phi : \mathscr{X} \to \mathscr{Y}$ (fonction cible, que l'on ne connaît pas !) et qui peut être approchée par une fonction $f \in \mathscr{F}$.

Tâche de l'apprentissage supervisé :
trouver une hypothèse $f \in \mathscr{F}$ qui approche *au mieux* la fonction cible ϕ.

Solution optimale à un problème d'apprentissage supervisé

Pour réaliser la tâche posée dans le problème d'apprentissage, il nous faut

1. **Fonction de coût** pour quantifier la qualité d'une hypothèse candidate (et potentiellement déterminer si elle est optimale)
2. **Méthode d'optimisation** un algorithme qui permet de trouver (efficacement) une hypothèse optimale (au sens de la fonction de coût) dans l'espace des hypothèses $\mathscr{F}$

Fonction de coût

Définition 2.7 Une fonction de coût

$$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$$
$$(y, f(\vec{x})) \mapsto L(y, f(\vec{x}))$$

aussi appelée fonction de perte ou fonction d'erreur (cost function or **L**oss function) est une fonction utilisée pour quantifier la qualité d'une prédiction.

La valeur $L(y, f(\vec{x}))$ est d'autant plus grande que l'étiquette prédite $f(\vec{x})$ est éloignée de la vraie valeur y.

Notion de risque

But de l'optimisation : trouver un f qui minimise ce coût sur l'ensemble des valeurs possibles de $\vec{x} \in \mathcal{X}$.

Définition 2.8 On appelle *risque* d'une hypothèse h [une fonction d'étiquetage arbitraire dans $\mathcal{F}$], l'espérance (sur toutes les observations de l'espace $\mathcal{X}$) d'une fonction de coût L :

$$\mathscr{R}(h) = \mathbb{E}_{\mathcal{X}}\left[L(h(\vec{x}), y)\right].$$

Avec cette définition, la fonction d'étiquetage f cherchée vérifie :

$$f = \arg\min_{h \in \mathcal{F}} \mathbb{E}\left[L(h(\vec{x}), y)\right]$$

Note : le risque ne peut généralement pas être calculé, mais, étant donné n observations étiquetées $\left\{\vec{x}^i, y^i\right\}_{i=1,\ldots,n}$ on approchera le risque par son estimation sur ces données observées :

suite...

. . . le Risque Empirique

Définition 2.9 On appelle *risque empirique* l'estimateur :

$$\mathscr{R}_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(\vec{x}^i), y^i).$$

Le prédicteur par minimisation du risque empirique est donc

$$f = \arg\min_{h\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n} L(h(\vec{x}^i), y^i).$$

Notes :

- pour certains choix de $\mathscr{F}$ le problème de minimisation du risque empirique peut avoir une solution analytique
- problème mal posé en général : pas de solution unique garantie, de multiples (infinité) de solutions peuvent minimiser le risque empirique.

Un problème mal posé : multiples (infinité de) solutions possibles !

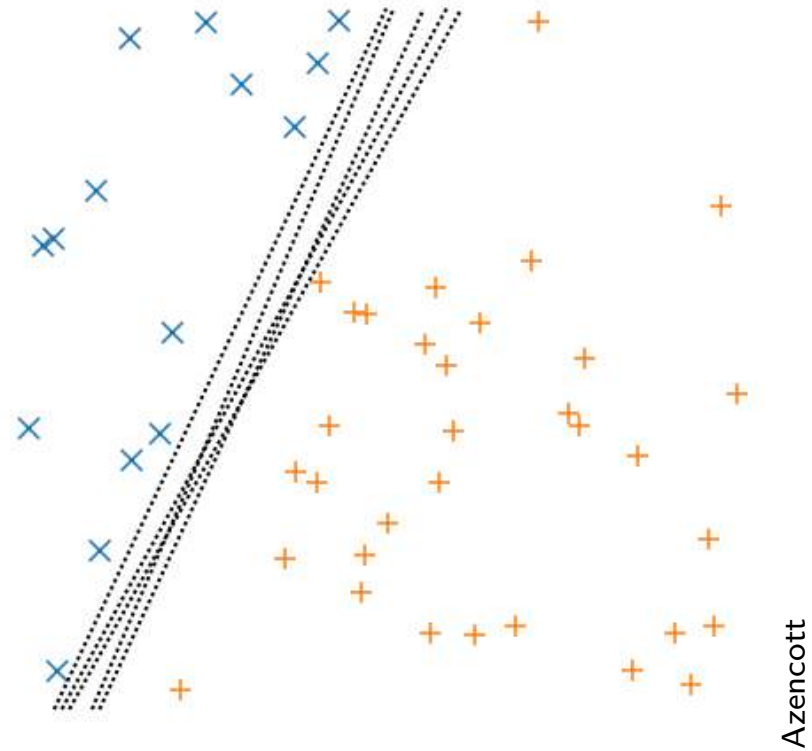
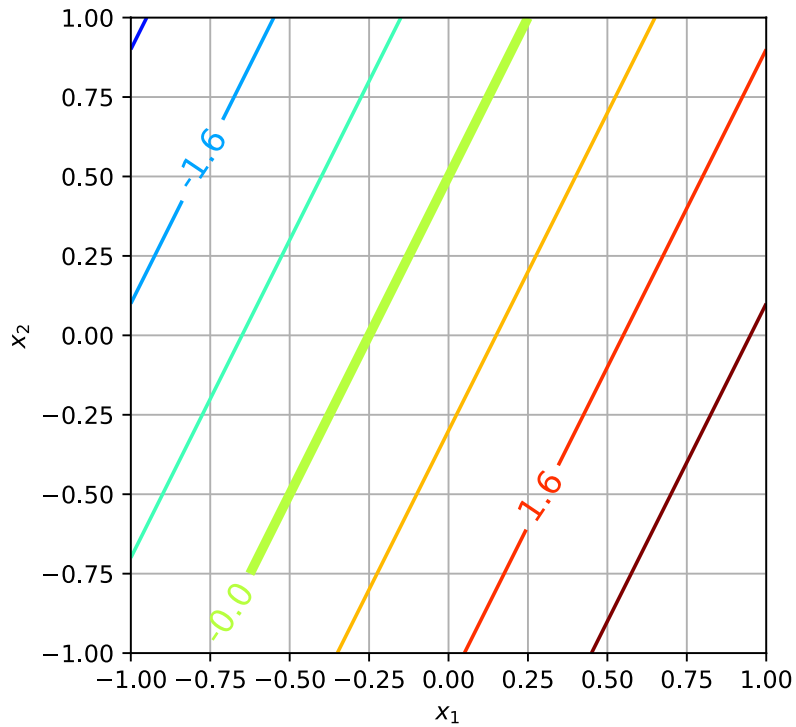


Figure 2.3 Une infinité de droites séparent parfaitement les points positifs (+) des points négatifs (x). Chacune d'entre elles a un risque empirique nul.

$$\mathscr{F} = \{\vec{x} \mapsto ax_1 + b - x_2\}$$



$$g(x_1, x_2) = ax_1 + b - x_2$$
$$a = 2,\ b = 0.5$$

Fonctions de coût

Grand choix de fonctions de coût. Comment choisir ?

- la fonction de coût est-elle adaptée au problème ?
- le problème d'optimisation résultant du coût choisi peut-il être résolu ?

Coût 0/1 pour la classification binaire

Définition 2.10 Coût 0/1 pour la classification binaire : Dans le cas d'une fonction f à valeurs binaires, on appelle *fonction de coût 0/1* (en anglais : 0/1 loss), la fonction :
Cas $\mathcal{Y} = \{0, 1\}$:

$$L_{0/1} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$$

$$y, f(\vec{x}) \mapsto L_{0/1}(y, f(\vec{x})) = \begin{cases} 1 & \text{si } f(\vec{x}) \neq y \\ 0 & \text{sinon.} \end{cases}$$

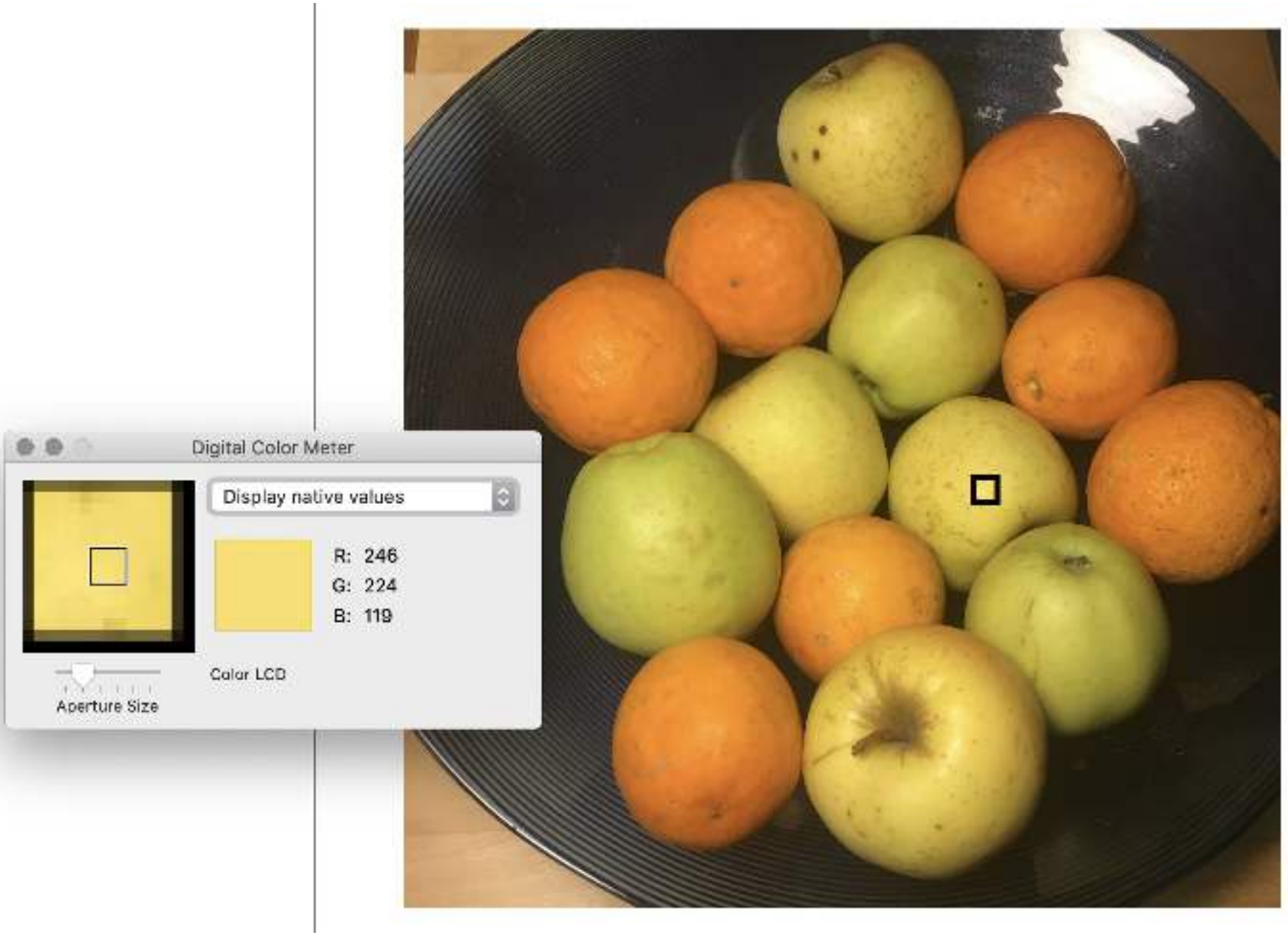
Cas $\mathcal{Y} = \{-1, 1\}$:

$$L_{0/1} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$$

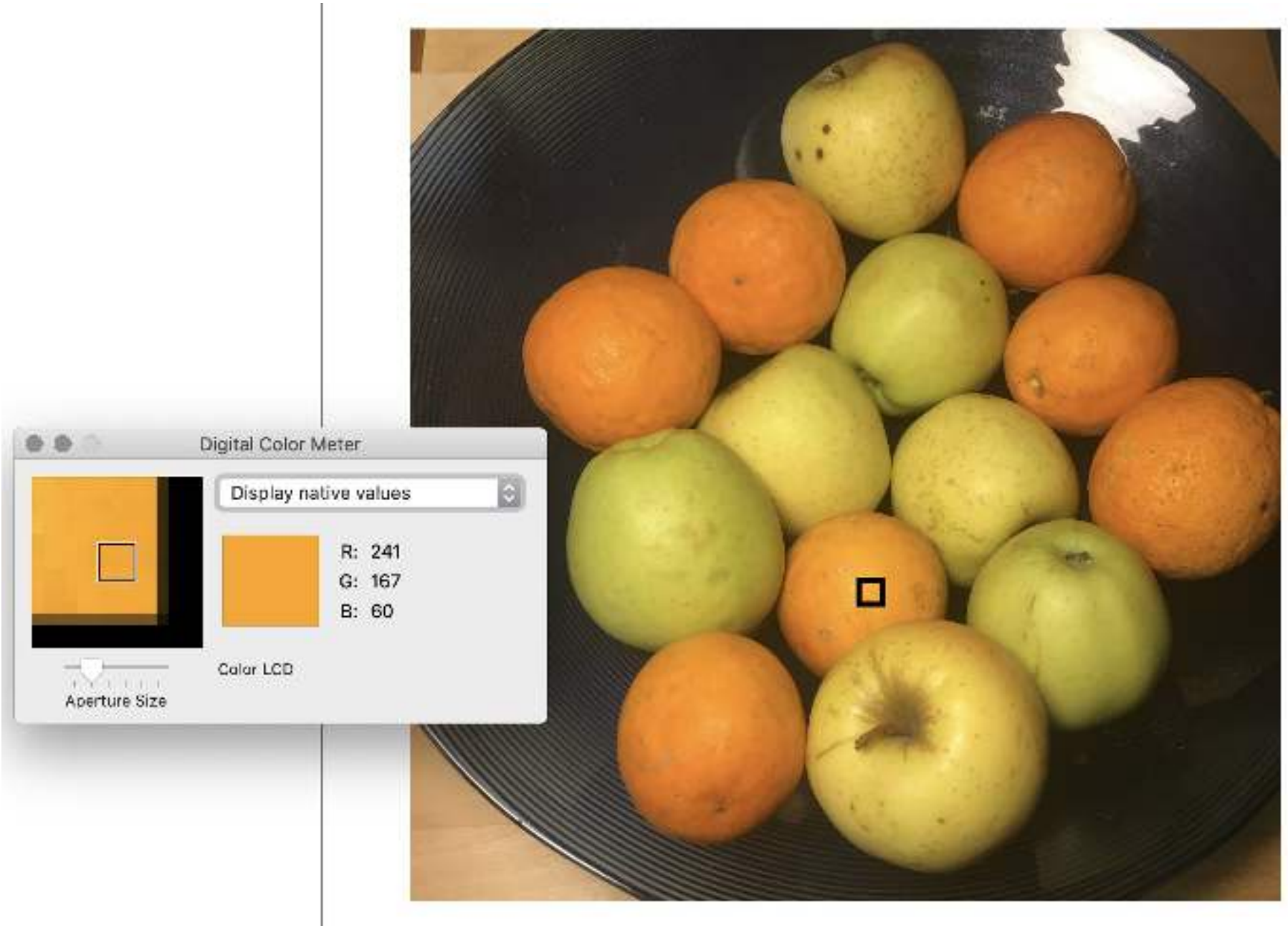
$$y, f(\vec{x}) \mapsto L_{0/1}(y, f(\vec{x})) = \frac{1 - y\, f(\vec{x})}{2}$$

Interprétation : avec cette fonction coût, le risque empirique est la proportion moyenne d'erreurs de prédiction sur le jeu d'entraînement.

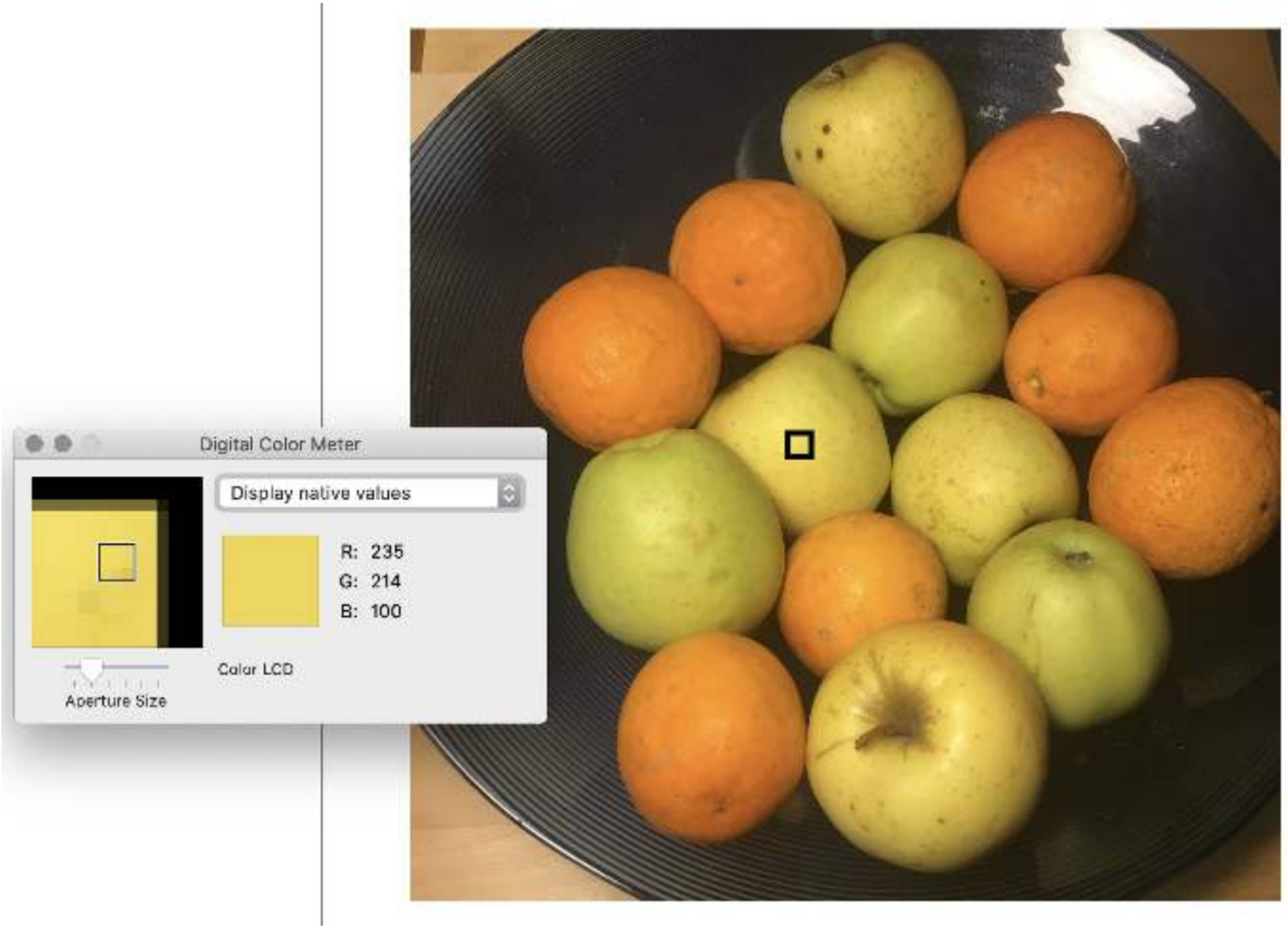
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 1$)



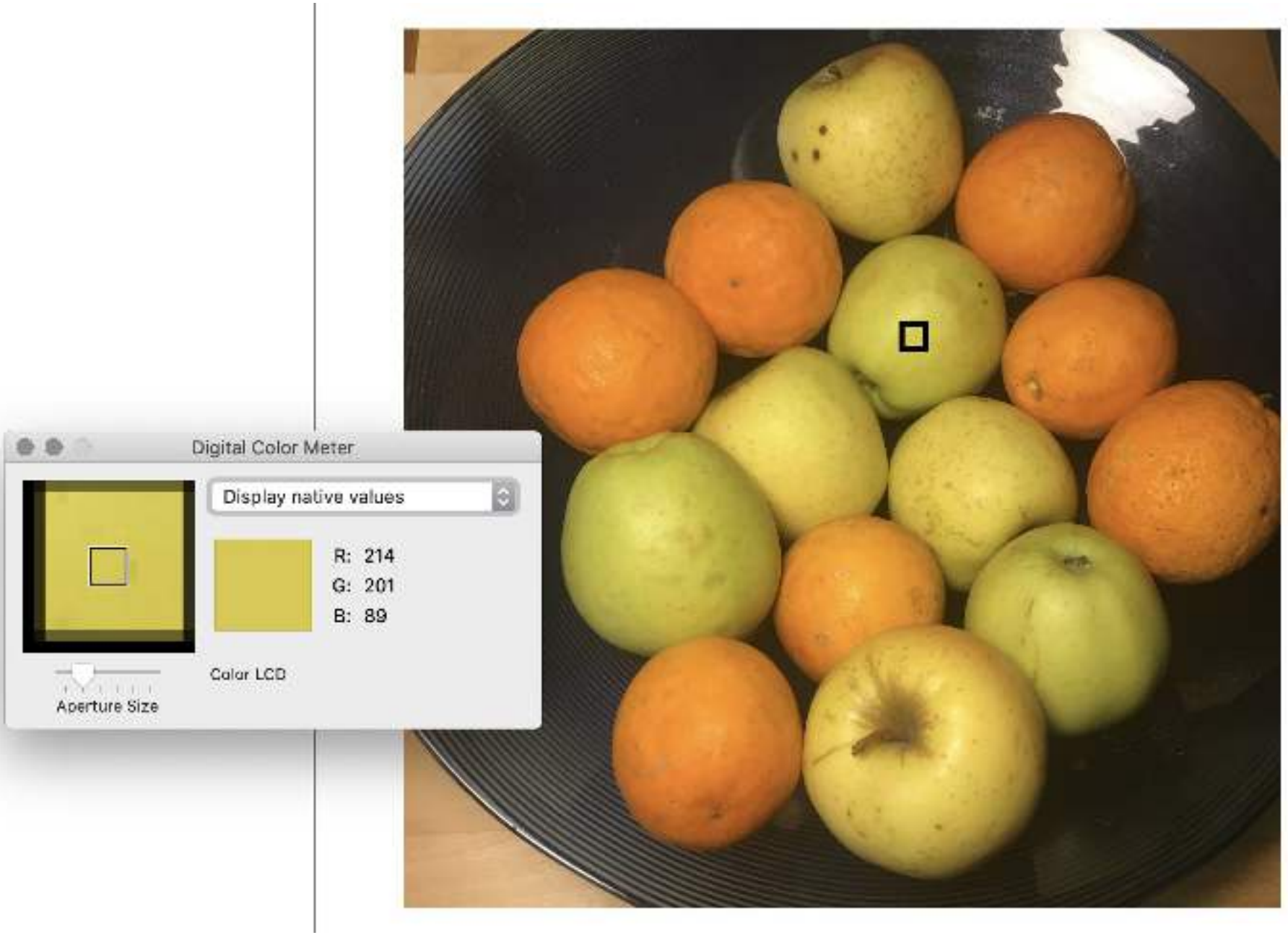
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 2$)



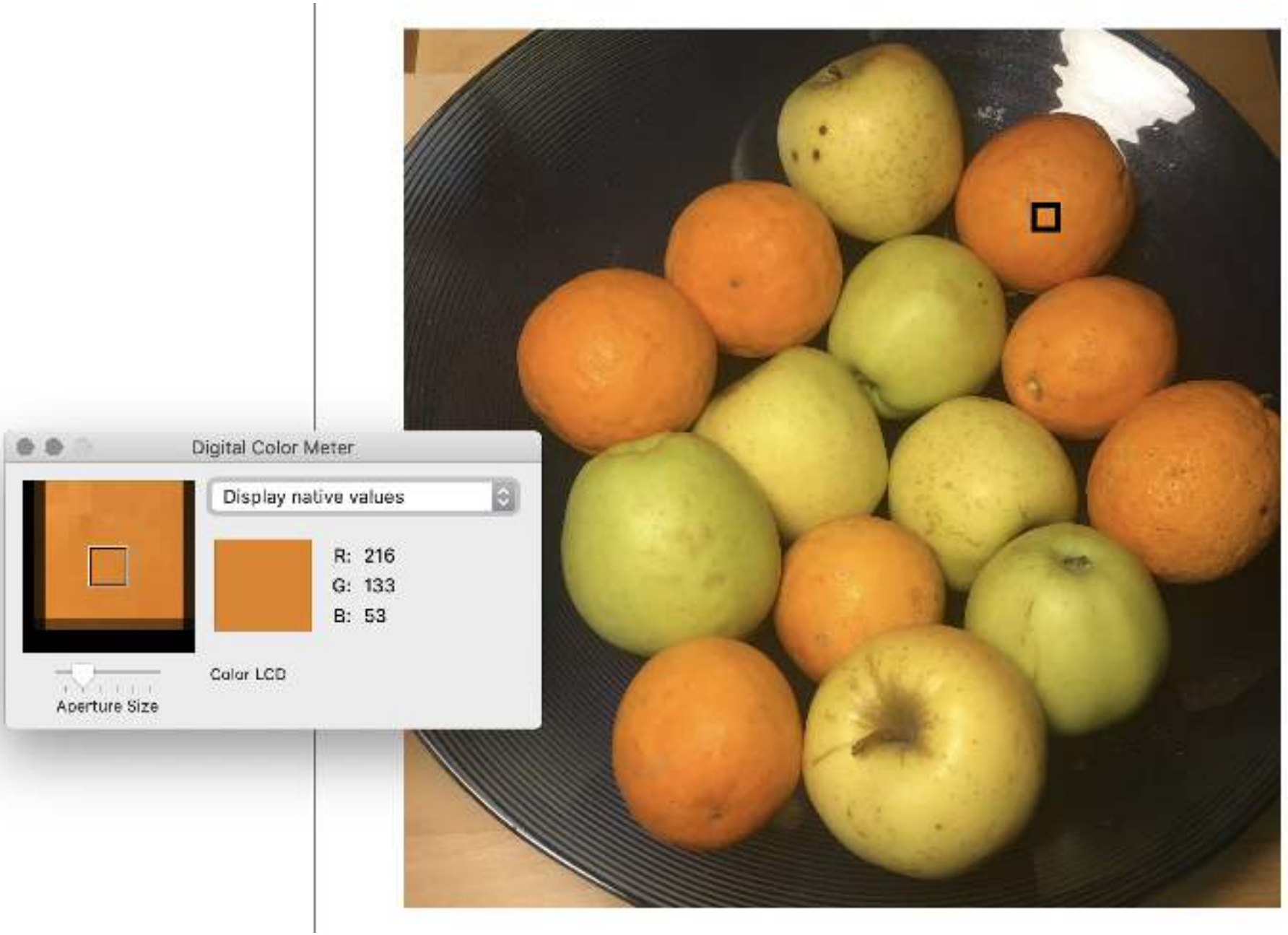
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 3$)



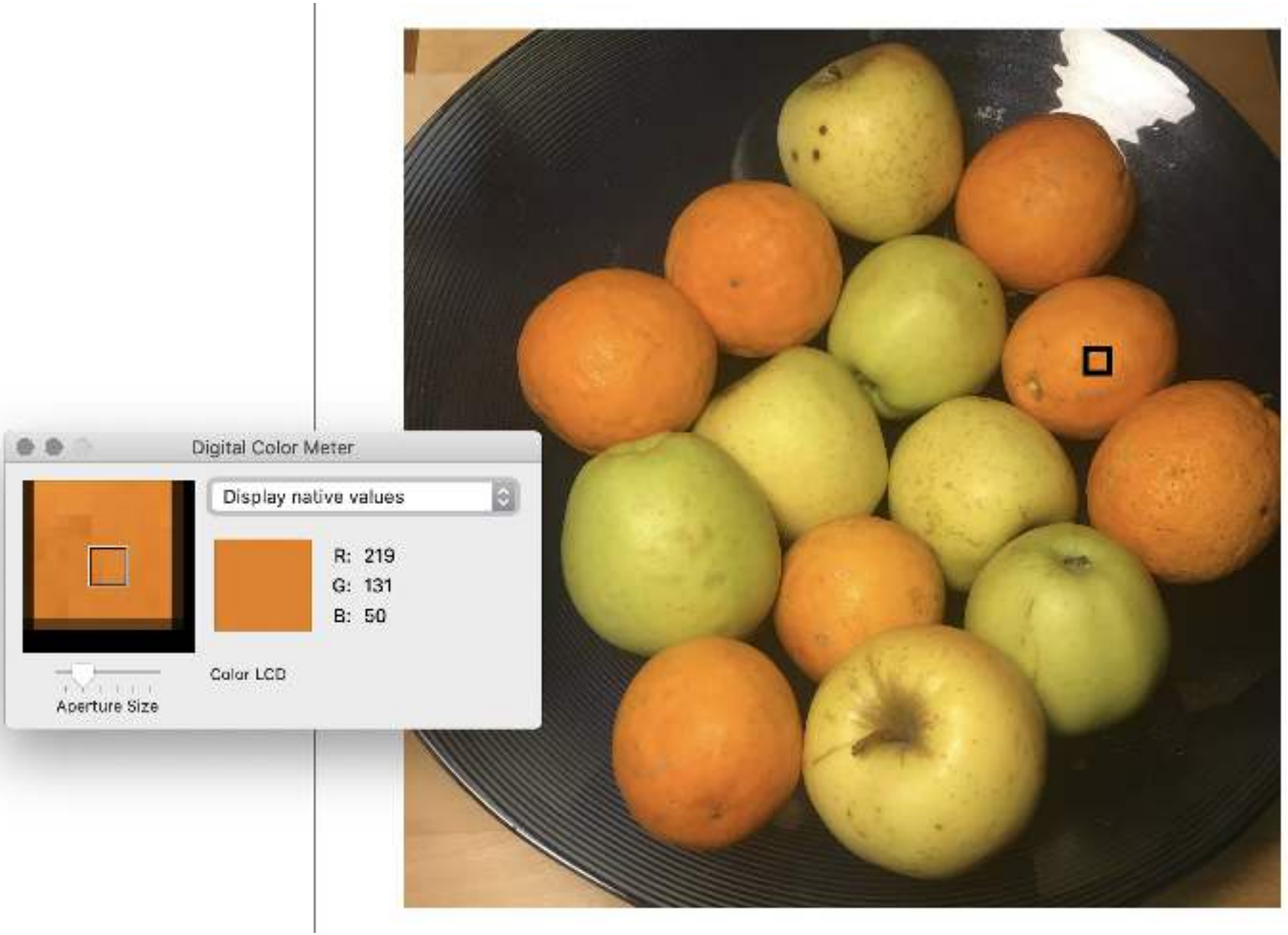
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 4$)



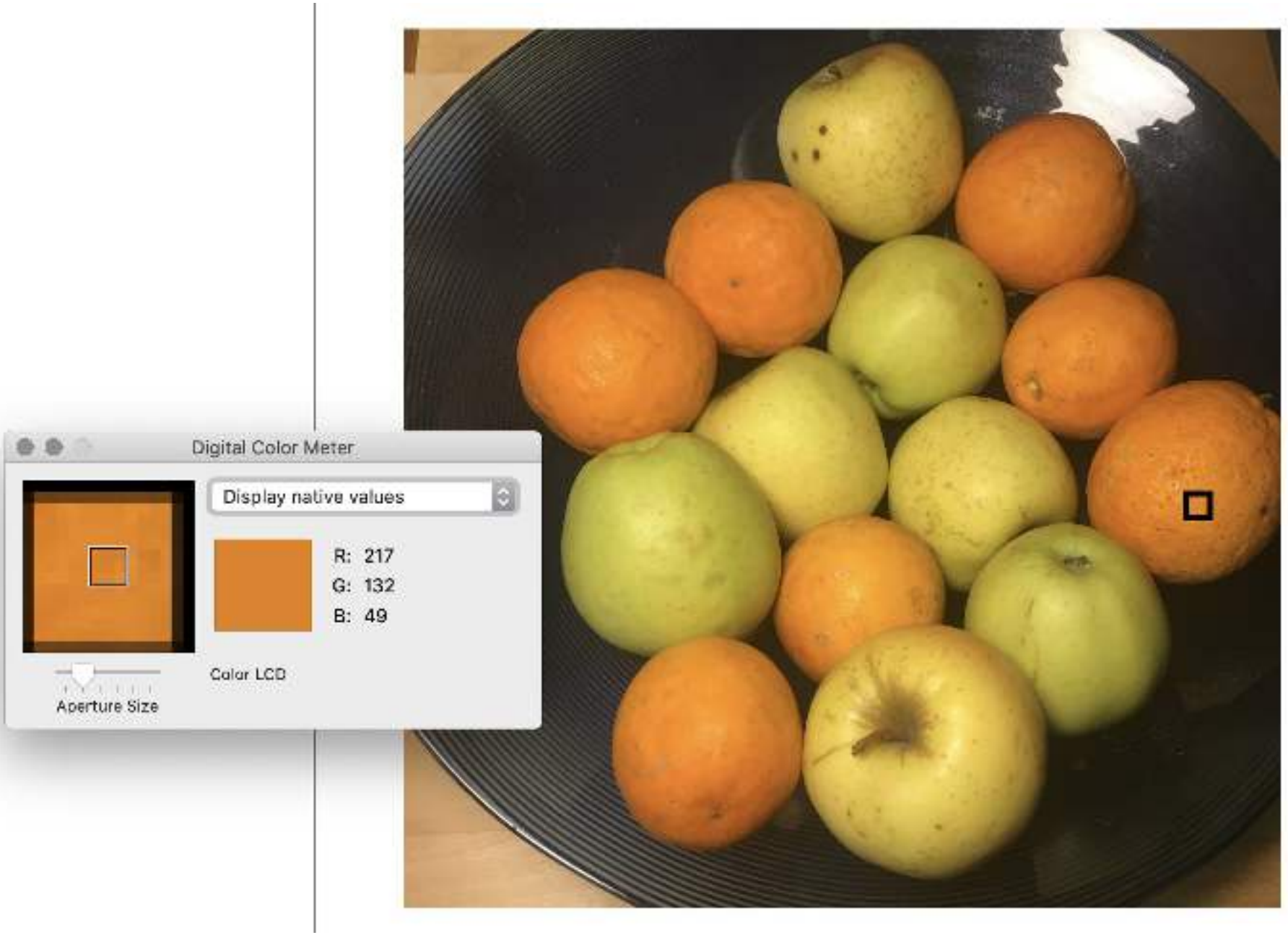
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 5$)



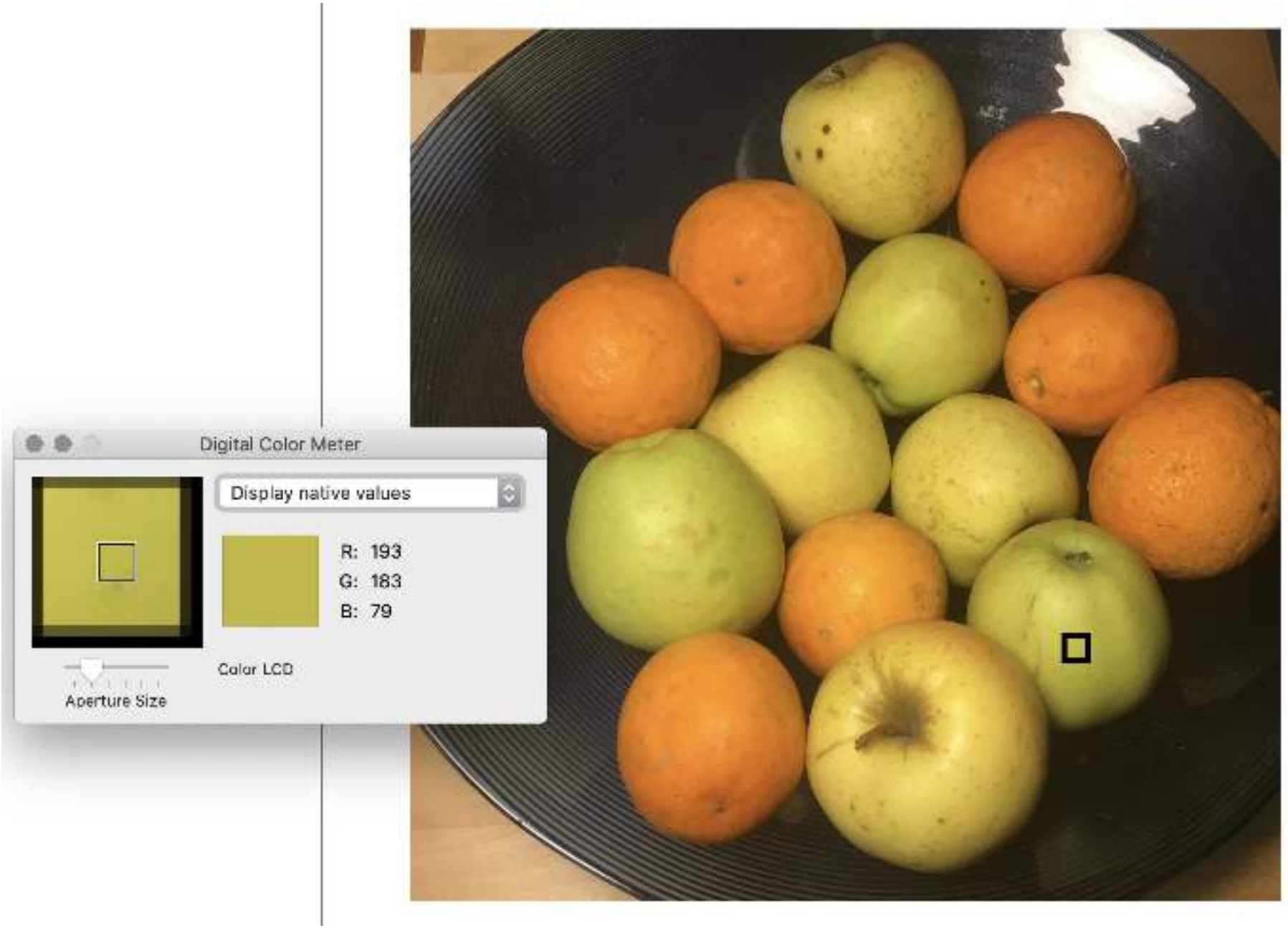
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 6$)



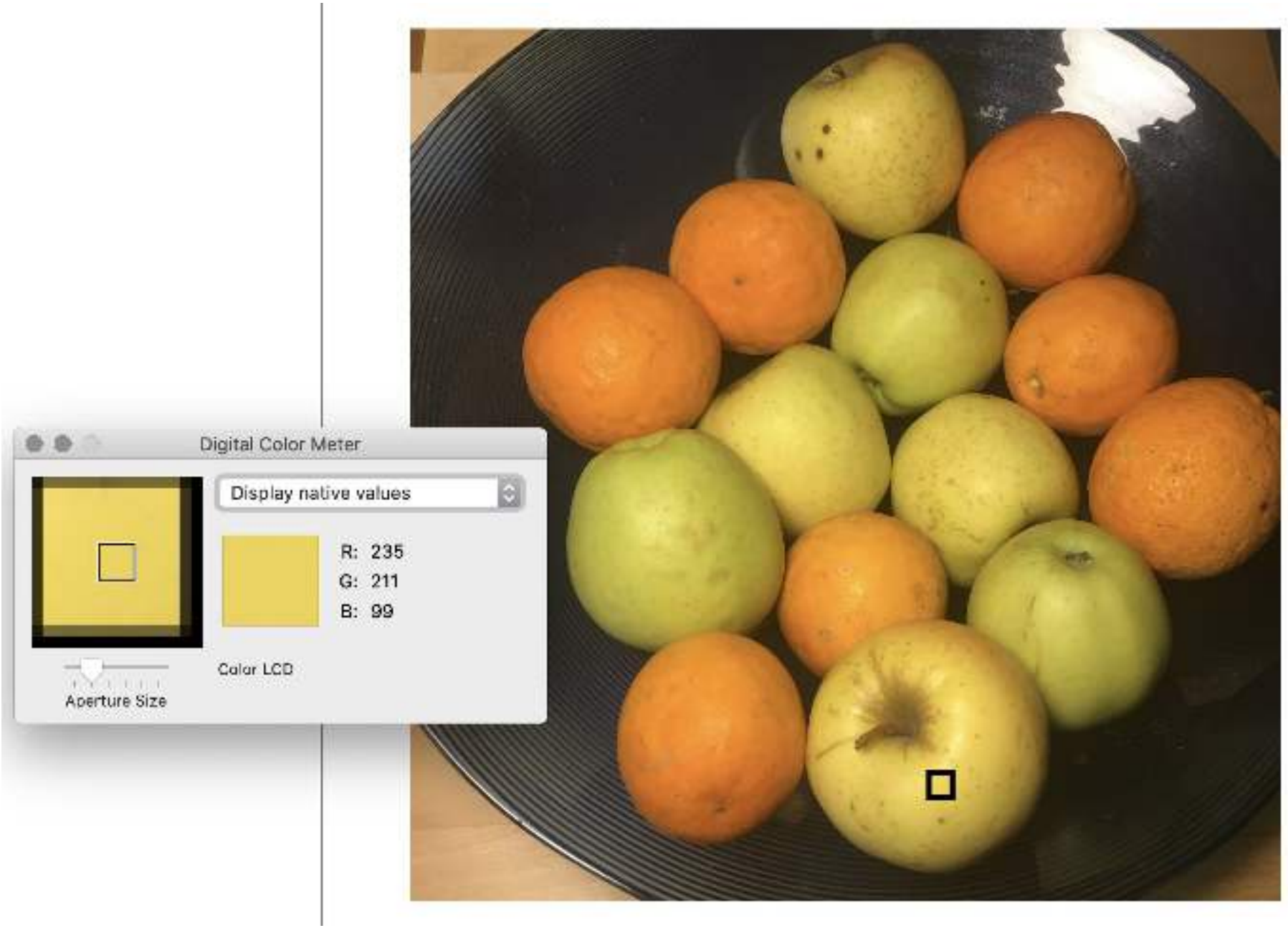
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 7$)



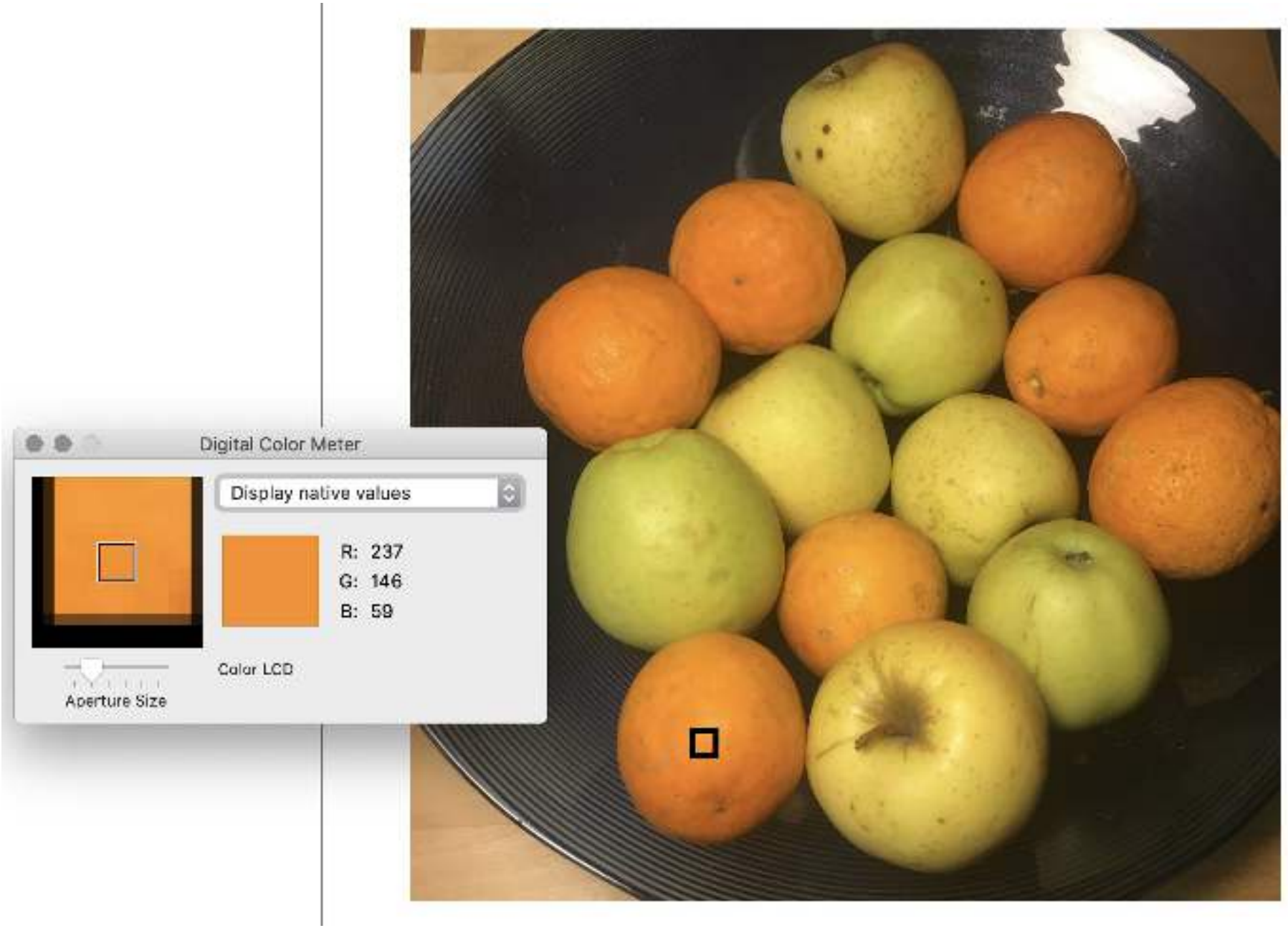
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 8$)



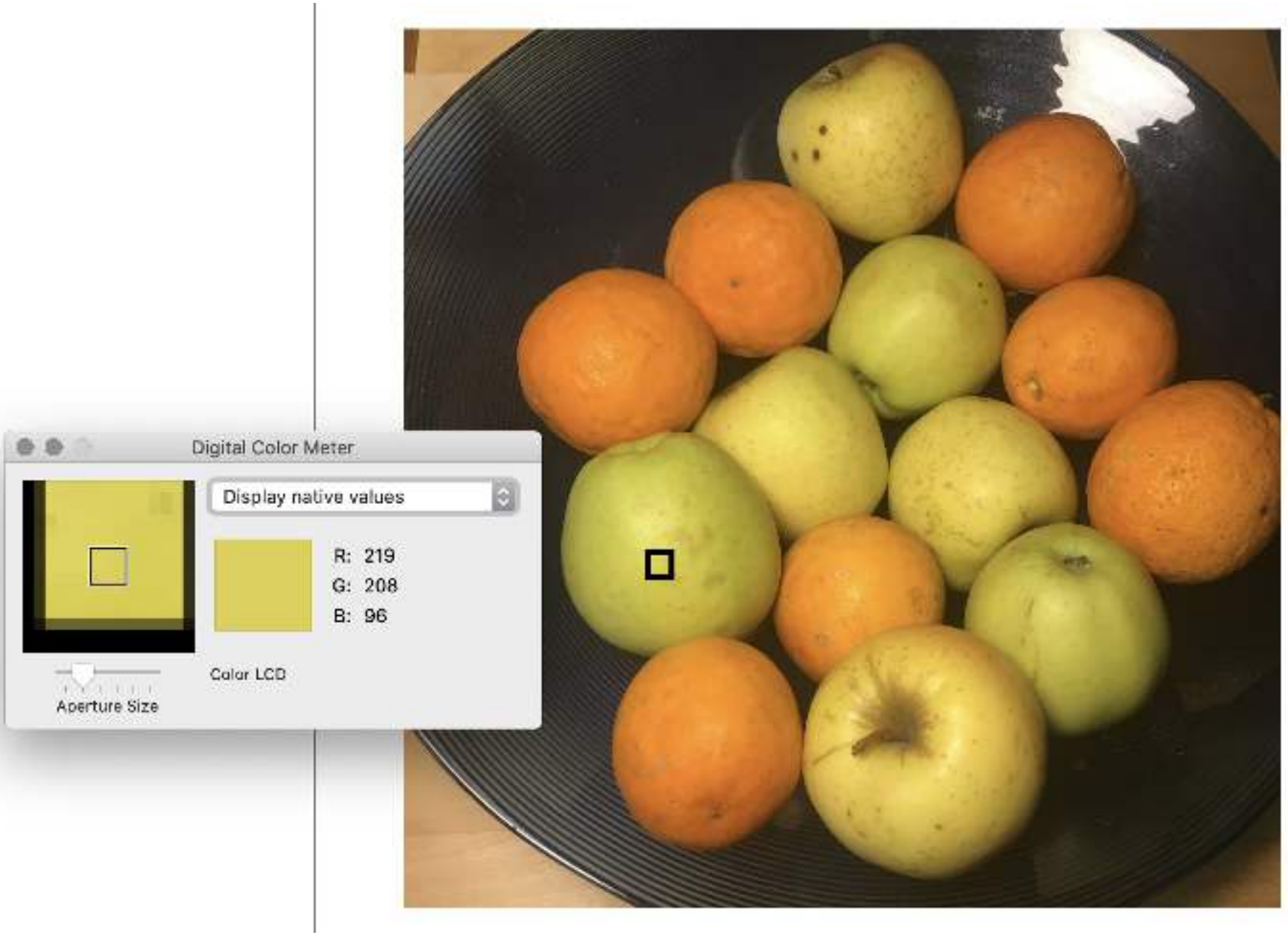
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 9$)



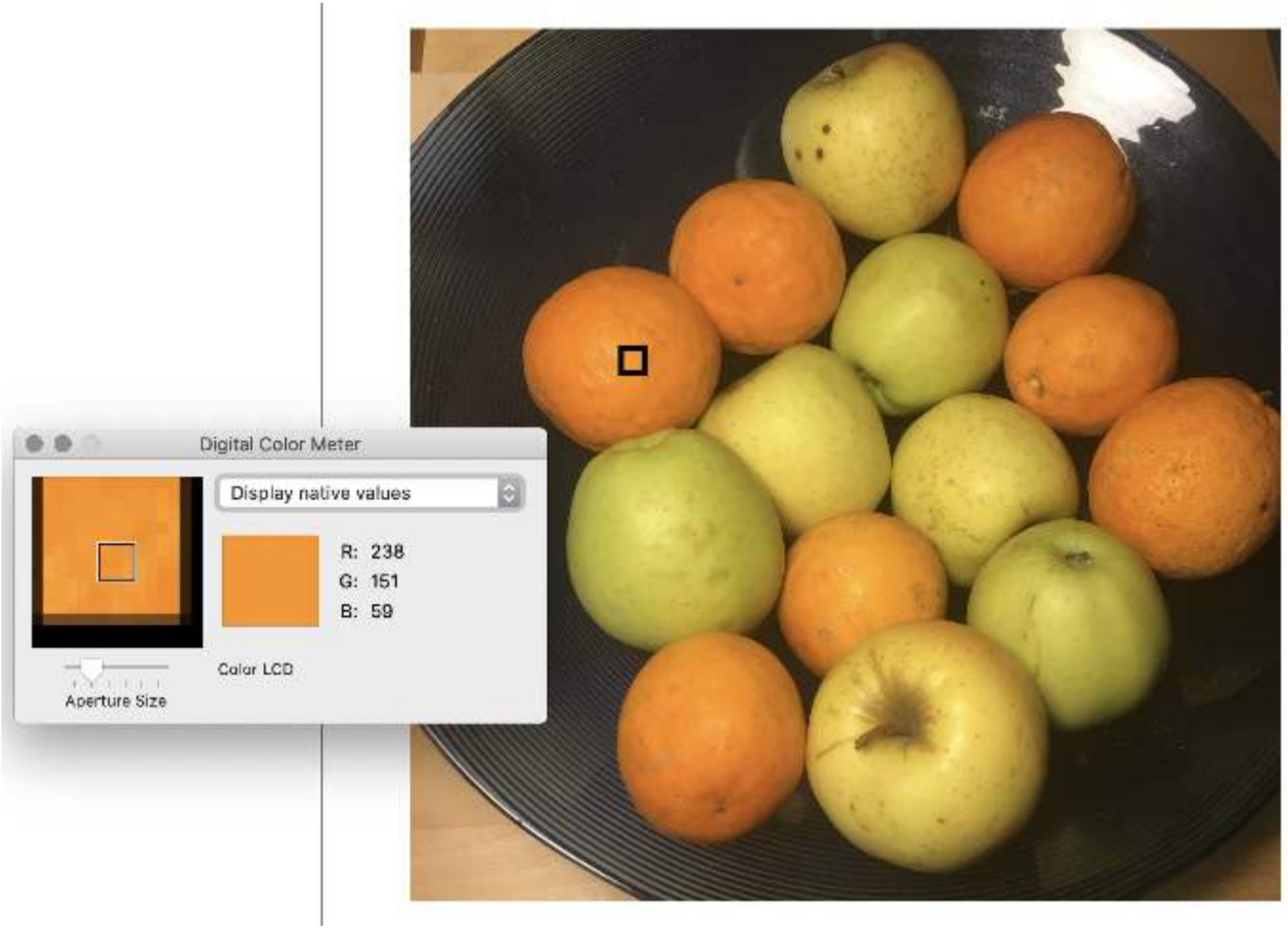
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 10$)



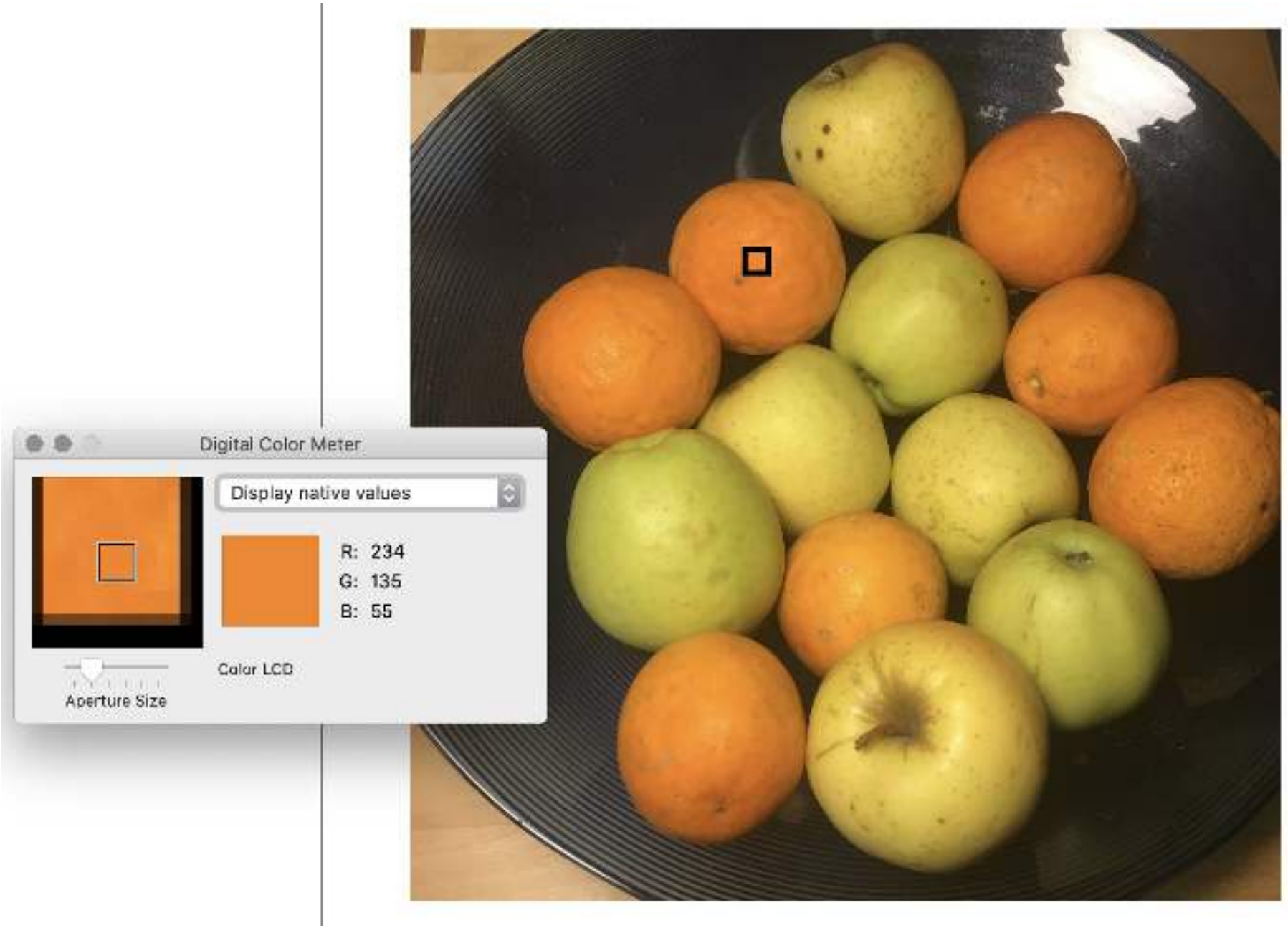
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 11$)



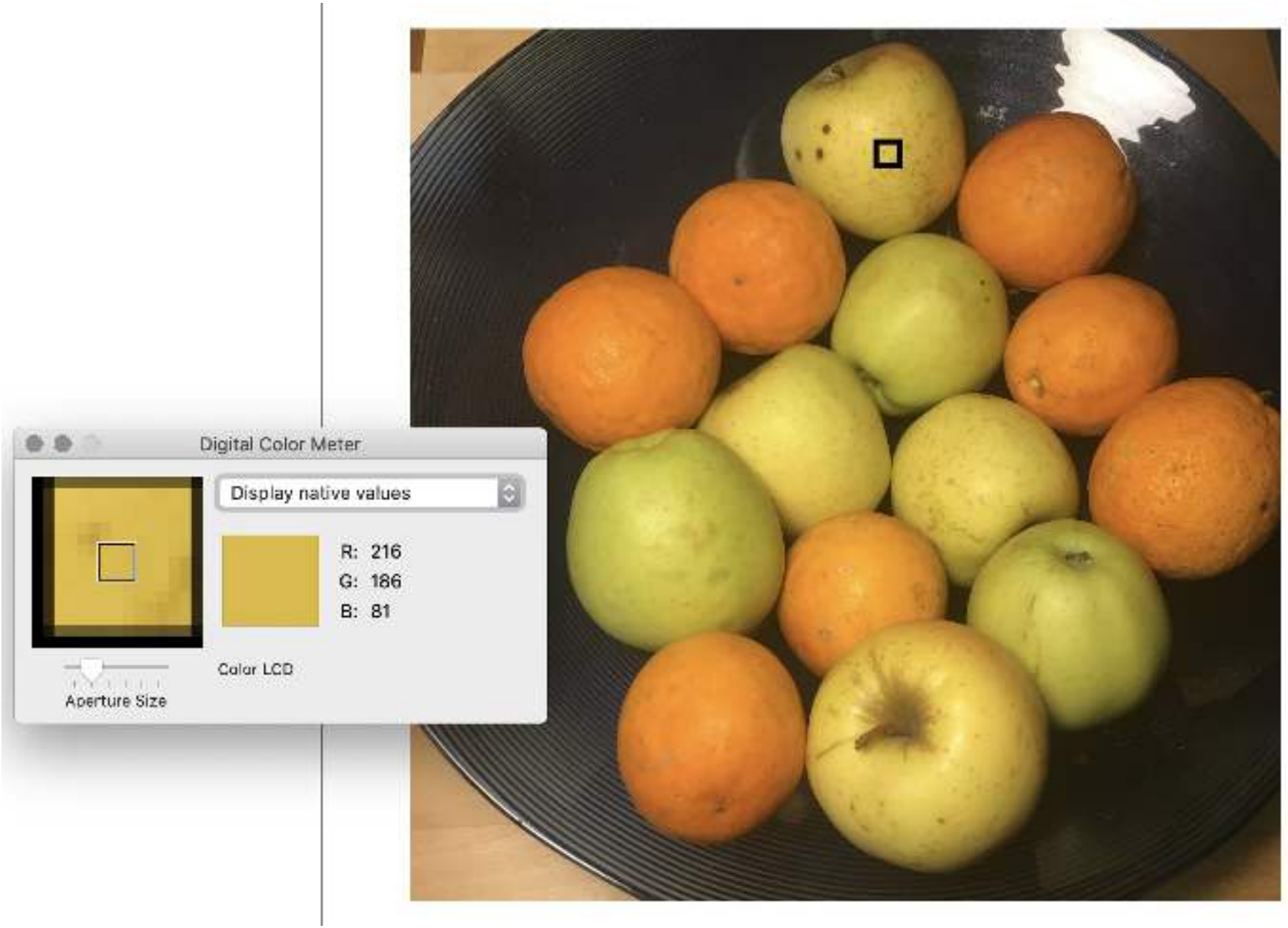
Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 12$)



Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 13$)



Pommes et oranges : valeurs RGB (Rouge-Vert-Bleu ; $i = 14$)



Classification pommes/oranges sur la base des valeurs RGB

$\vec{x}^i = (x_1^i, x_2^i, x_3^i)$ (=RGB observé)	y^i (étiquette)	classe	index
$\vec{x}^1 = (246 \quad 224 \quad 119)$	$y^1 = 1$	pomme	$i = 1$
$\vec{x}^2 = (241 \quad 167 \quad 60)$	$y^2 = -1$	orange	$i = 2$
$\vec{x}^3 = (235 \quad 214 \quad 100)$	$y^3 = 1$	pomme	$i = 3$
$\vec{x}^4 = (214 \quad 201 \quad 89)$	$y^4 = 1$	pomme	$i = 4$
$\vec{x}^5 = (216 \quad 133 \quad 53)$	$y^6 = -1$	orange	$i = 5$
$\vec{x}^6 = (219 \quad 131 \quad 50)$	$y^6 = -1$	orange	$i = 6$
$\vec{x}^7 = (217 \quad 132 \quad 49)$	$y^7 = -1$	orange	$i = 7$
$\vec{x}^8 = (193 \quad 183 \quad 79)$	$y^8 = 1$	pomme	$i = 8$
$\vec{x}^9 = (235 \quad 211 \quad 99)$	$y^9 = 1$	pomme	$i = 9$
$\vec{x}^{10} = (237 \quad 146 \quad 59)$	$y^{10} = -1$	orange	$i = 10$
$\vec{x}^{11} = (219 \quad 208 \quad 96)$	$y^{11} = 1$	pomme	$i = 11$
$\vec{x}^{12} = (238 \quad 151 \quad 59)$	$y^{12} = -1$	orange	$i = 12$
$\vec{x}^{13} = (234 \quad 135 \quad 55)$	$y^{13} = -1$	orange	$i = 13$
$\vec{x}^{14} = (216 \quad 186 \quad 81)$	$y^{14} = 1$	pomme	$i = 14$

Considérons comme fonction de classification : $f(\vec{x}) = f_T(\vec{x}) =$

$$\begin{cases} +1 \quad \text{(pomme)} & \text{si } g(\vec{x}) = x_3 - T > 0 \quad \text{(forte contribution de bleu)} \\ -1 \quad \text{(orange)} & \text{si } g(\vec{x}) = x_3 - T \leq 0 \quad \text{(faible contribution de bleu)} \end{cases}$$

Risque empirique

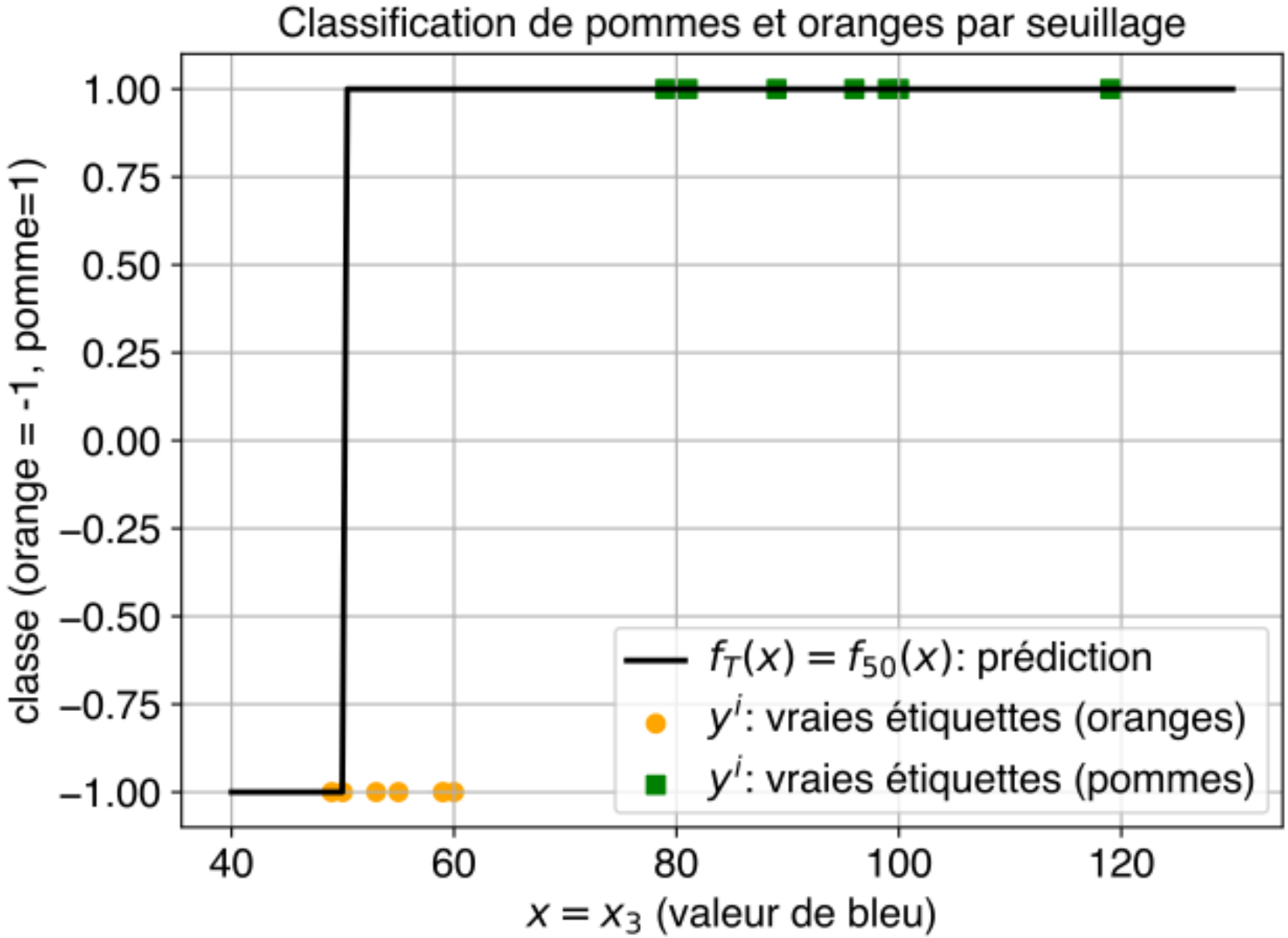
La valeur de T influence le risque empirique :

$$\mathscr{R}_n(f_T) = \frac{1}{n}\sum_{i=1}^{n} L_{0/1}(f_T(\vec{x}^i), y^i).$$

Essayons le seuil $T = 50$:

i	x_2^i	$f(\vec{x}) = f_{50}(x_2^i)$	y^i	$f(\vec{x}^i)\, y^i$	$L_{0/1}(f_{50}((\vec{x}^i), y^i))$
1	119	1	1	1	0
2	60	1	-1	-1	1
3	100	1	1	1	0
4	89	1	1	1	0
5	53	1	-1	-1	1
6	50	-1	-1	1	0
7	49	-1	-1	1	0
8	79	1	1	1	0
9	99	1	1	1	0
10	59	1	-1	-1	1
11	96	1	1	1	0
12	59	1	-1	-1	1
13	55	1	-1	-1	1
14	82	1	1	1	0
					$\mathscr{R}_{14}(f_{50}) = 5/14$

Classification avec fonction de décision seuil $T = 50$



Risque empirique : essayons une autre valeur pour T

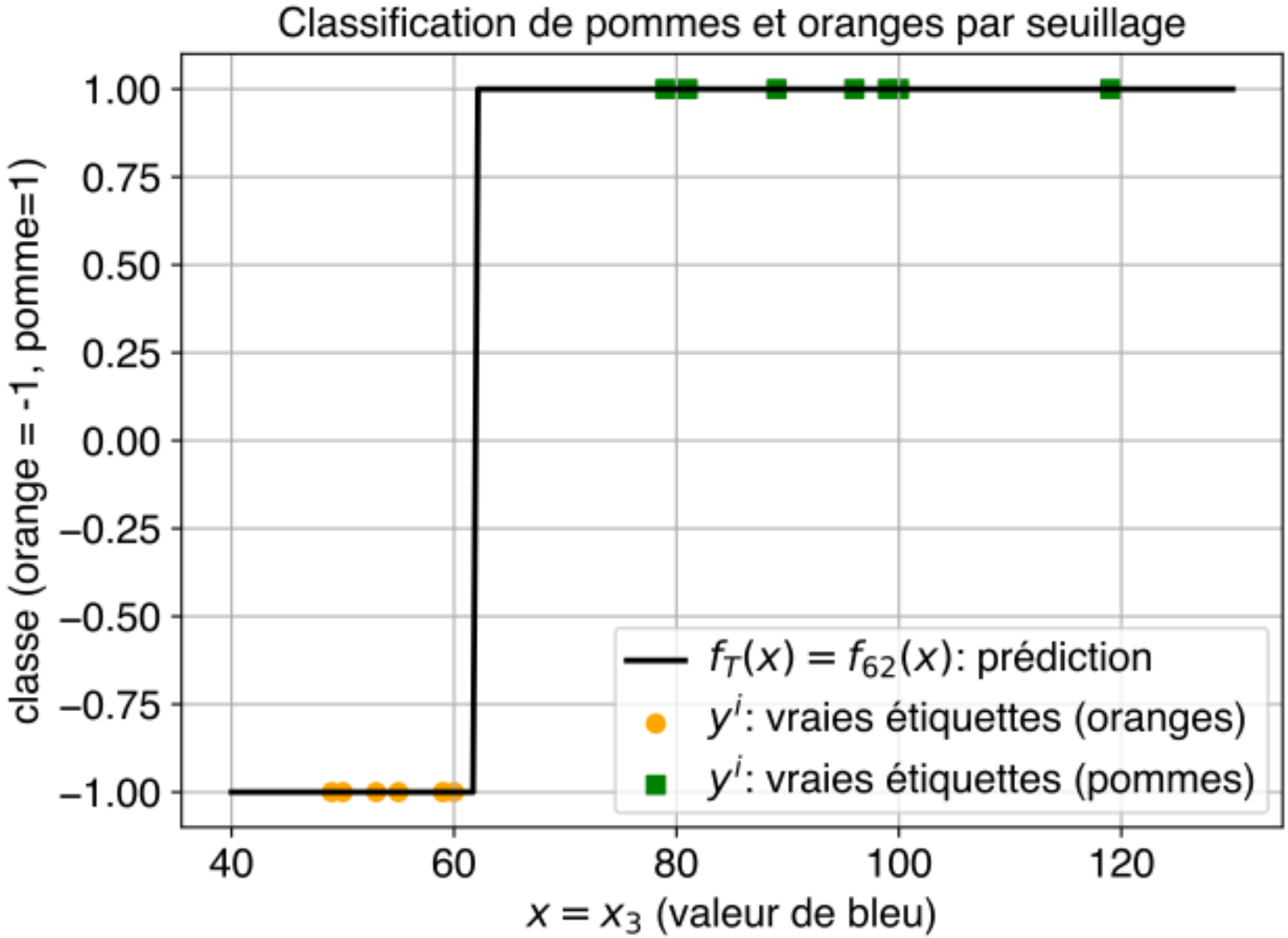
La valeur de T influence le risque empirique :

$$\mathscr{R}_n(f_T) = \frac{1}{n}\sum_{i=1}^{n} L_{0/1}(f_T(\vec{x}^i), y^i).$$

Essayons le seuil $T = 62$:

i	x_2^i	$f(\vec{x}) = f_{62}(x_2^i)$	y^i	$f(\vec{x}^i)\, y^i$	$L_{0/1}(f_{62}((\vec{x}^i), y^i)$
1	119	1	1	1	0
2	60	-1	-1	1	0
3	100	1	1	1	0
4	89	1	1	1	0
5	53	-1	-1	1	0
6	50	-1	-1	1	0
7	49	-1	-1	1	0
8	79	1	1	1	0
9	99	1	1	1	0
10	59	-1	-1	1	0
11	96	1	1	1	0
12	59	-1	-1	1	0
13	55	-1	-1	1	0
14	82	1	1	1	0
					$\mathscr{R}_{14}(f_{62}) = 0$

Classification avec fonction de décision seuil $T = 62$



Minimisation du risque empirique

Noter que :

- il existe (au moins) une solution $f_T(\vec{x})$ qui minimise le risque
- la solution n'est pas unique : une infinité de solutions minimisent le risque empirique, il suffit de prendre $T \in [60, 79)$.

Classification binaire par régression

On peut décider de déterminer une fonction de décision (à valeurs réelles) en faisant une régression sur les étiquettes.

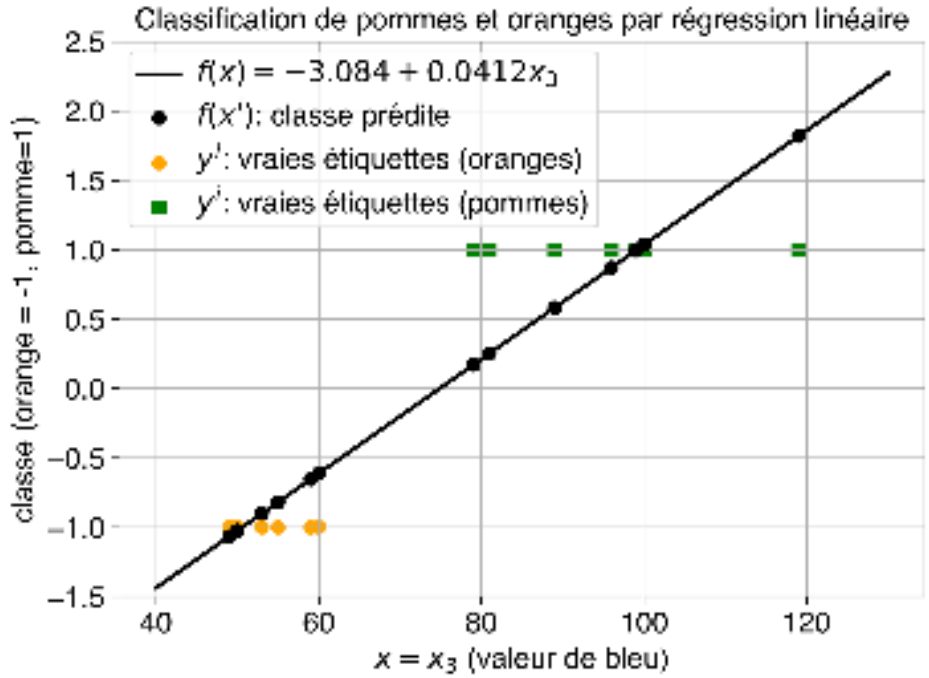
Exemple : Classification binaire par régression avec fonction de décision linéaire
On considère que la fonction à valeurs réelles est donnée par

$$f(\vec{x}) = f(x_2) = \beta_0 + \beta_1 x_2$$

Attention : f s'apparente ici à une fonction de décision (que nous avons dénotée g auparavant) et n'est plus directement la fonction de prédiction (ou fonction "étiquetage") qui ne prendrait que les valeurs binaires -1 et 1. L'étiquette prédite est celle dont la valeur est la plus proche de la valeur retournée par $f(\vec{x})$.

Régression linéaire pour classification binaire

$$f(\vec{x}) = f(x_2) = \beta_0 + \beta_1 x_2$$



Note : $f(\vec{x})$ n'est pas binaire ! On peut dire que la proximité aux valeurs -1 ou 1 indique directement la classe.

Questions :

- Quelle est la qualité de cette approximation ?
- Comment trouver β_0 et β_1 ?

Fonctions de perte pour les fonctions de décision à valeurs réelles pour la classification binaire

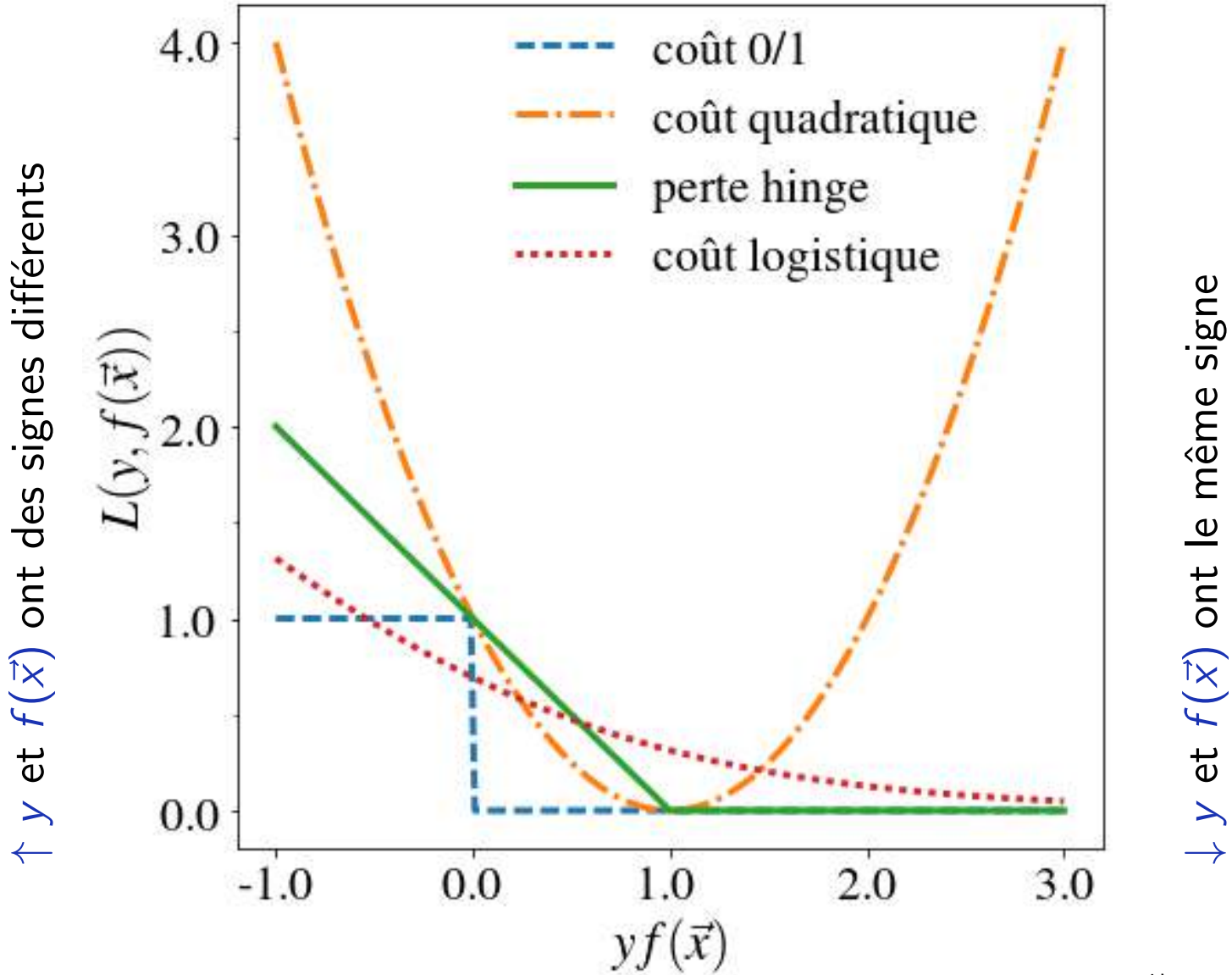


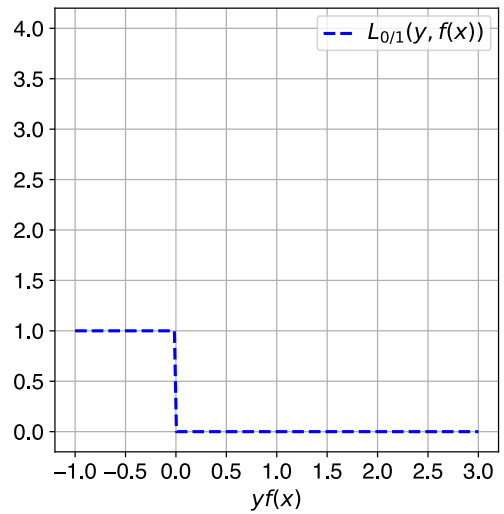
FIGURE 2.4 – Fonctions de perte pour la classification binaire.

Azencott

Coût 0/1 pour fonctions de décision à valeurs réelles (classification binaire par régression)

Définition 2.11 On appelle fonction de coût 0/1 pour fonction de décision f à valeurs réelles ($\mathcal{Y} = \{-1, 1\}$), la fonction :
$L_{0/1} : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}$

$$y, f(\vec{x}) \mapsto \begin{cases} 1 & \text{si } yf(\vec{x}) \leq 0 \\ 0 & \text{sinon} \end{cases}$$

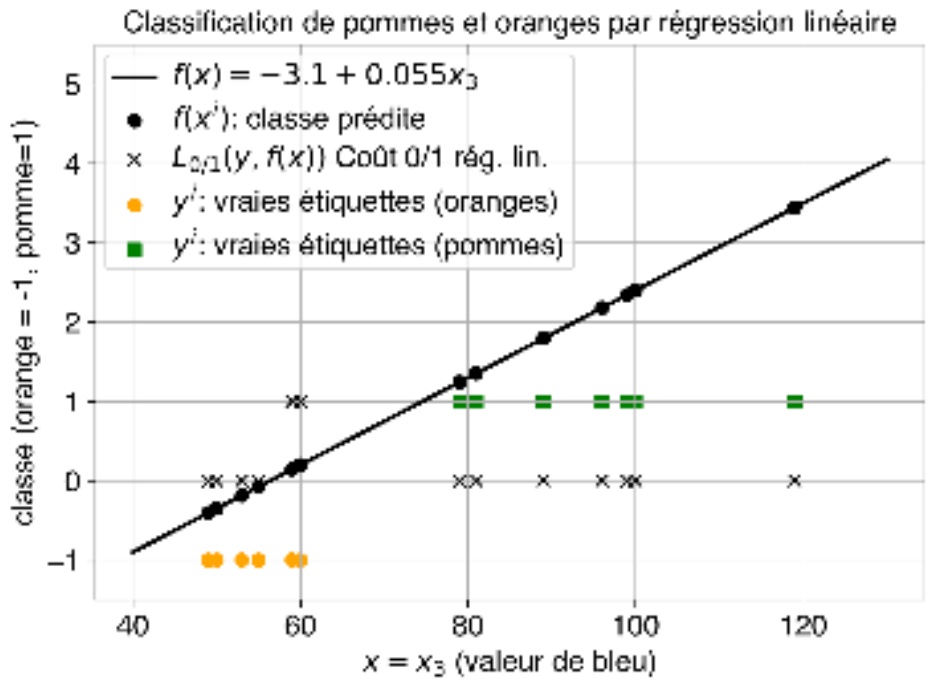


Remarques

- On regarde le signe du produit $yf(\vec{x})$ et si les signes concordent : pas de pénalité
- attention : f dénote ici une fonction de décision (que nous avons dénotée g auparavant) et non pas la fonction de prédiction (ou fonction "étiquetage") qui ne prend que des valeurs binaires.
- la fonction de coût 0/1 n'est pas dérivable
- cette fonction est peu fine : elle ne distingue pas les écarts de $f(\vec{x})$ proches ou éloignés de l'étiquette y.

Coût 0/1 pour Régression linéaire pour classification binaire

$$f(\vec{x}) = f(x_2) = -3.1 + 0.055x_2$$



Le risque empirique avec le coût 0/1 est (nombre de croix ($\times$) pour lesquelles la valeur est 1, divisé par le nombre d'observations) :

$$\mathscr{R}_{14}(h) = \frac{1}{14}\sum_{i=1}^{14} L_{0/1}(f(\vec{x}^i), y^i) = \frac{3}{14}$$

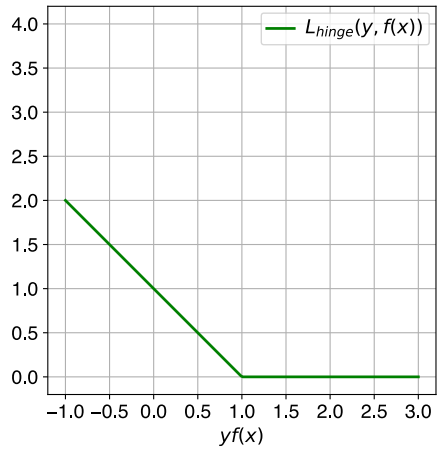
Note : il y a deux observations pour lesquelles $x_3 = 59$.

Erreur hinge pour la classification binaire

Définition 2.12 On appelle fonction d'erreur hinge, ou hinge loss, la fonction

$$L_{\text{hinge}} : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}$$

$$y, f(\vec{x}) \mapsto \begin{cases} 0 & \text{si } yf(\vec{x}) \geq 1 \\ 1 - yf(\vec{x}) & \text{sinon} \end{cases}$$



Notations équivalentes :

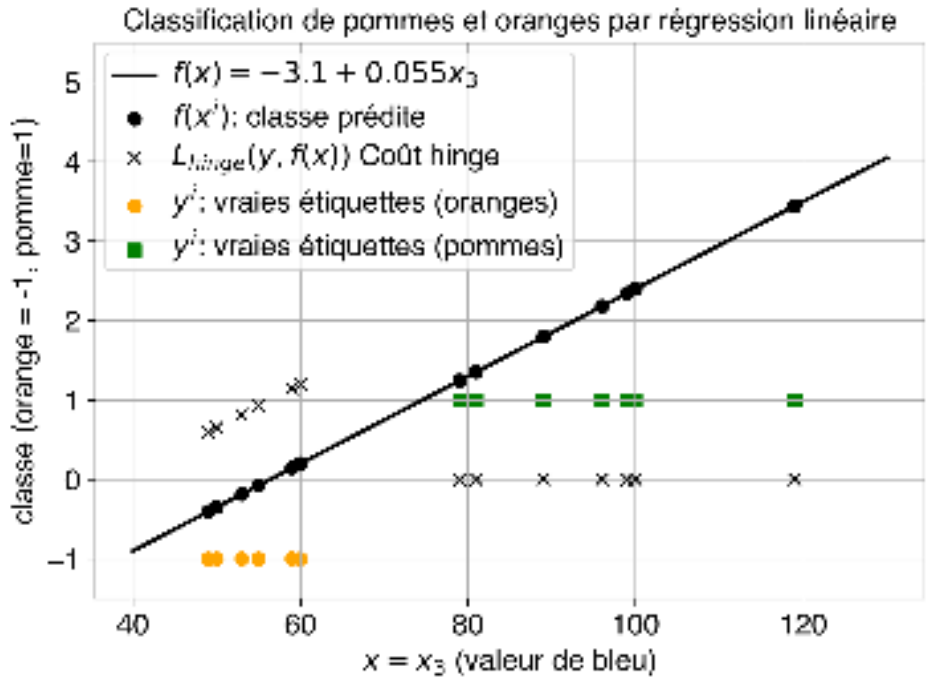
$$L_{\text{hinge}}(y, f(\vec{x})) = \max(0, 1 - yf(\vec{x})) = [1 - yf(\vec{x})]_+$$

Remarques

- pour une classification parfaite (quand $\mathcal{Y} = \{-1, 1\}$) on a $yf(\vec{x}) = 1$
- Fonction coût est d'autant plus grande que $yf(\vec{x})$ s'éloigne de 1 à gauche
- On considère qu'il n'y a pas d'erreur si $yf(\vec{x}) > 1$
- hinge = charnière ; aspect de coude

Coût hinge pour Régression linéaire pour classification binaire

$$f(\vec{x}) = f(x_2) = -3.1 + 0.055x_2$$



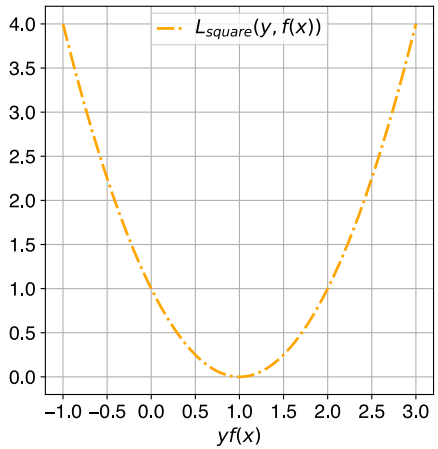
Le risque empirique avec le coût hinge est (somme des valeurs pour les croix ($\times$) divisée par le nombre d'observations) :

$$\mathscr{R}_{14}(h) = \frac{1}{14}\sum_{i=1}^{14} L_{\text{hinge}}(f(\vec{x}^i), y^i) = \frac{6.47}{14}$$

Coût quadratique pour la classification binaire

Définition 2.13 on appelle coût quadratique (square loss) la fonction

$$L_{\text{square}} : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto (1 - yf(\vec{x}))^2$$

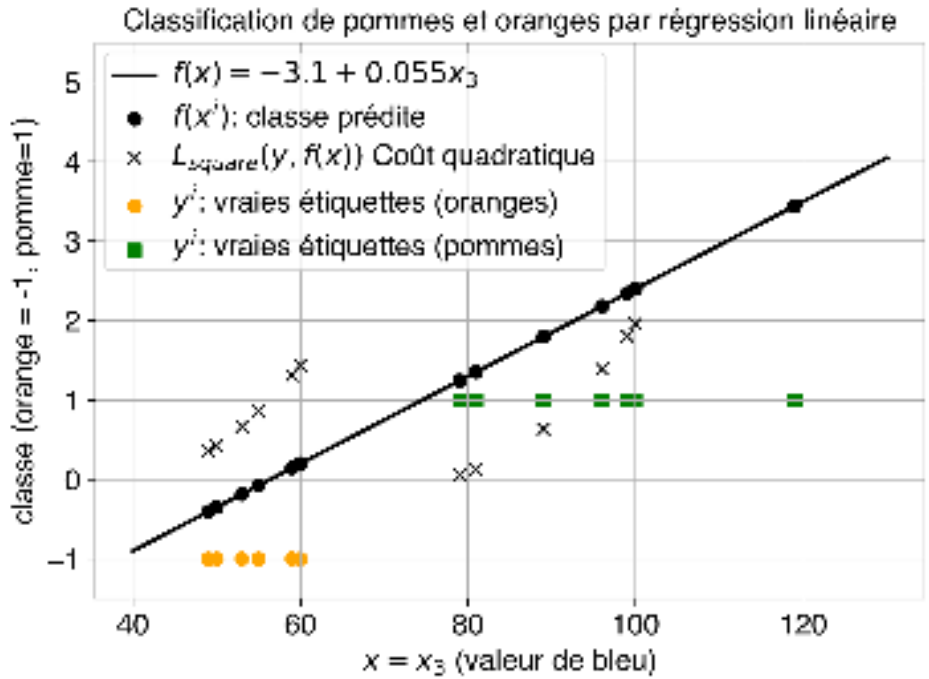


Remarques

- Ici, on veut favoriser $f(\vec{x})$ la plus proche possible de 1 pour les observations positives (et -1 pour les observations négatives). Ainsi, on pénalisera aussi les cas où $yf(\vec{x})$ s'éloigne de 1 par la droite.
- fonction de coût dérivable : on peut, dans certains cas trouver une solution analytique (e.g. pour la régression linéaire il y a une solution explicite)

Coût quadratique pour Régression linéaire pour classification binaire

$$f(\vec{x}) = f(x_2) = -3.1 + 0.055x_2$$

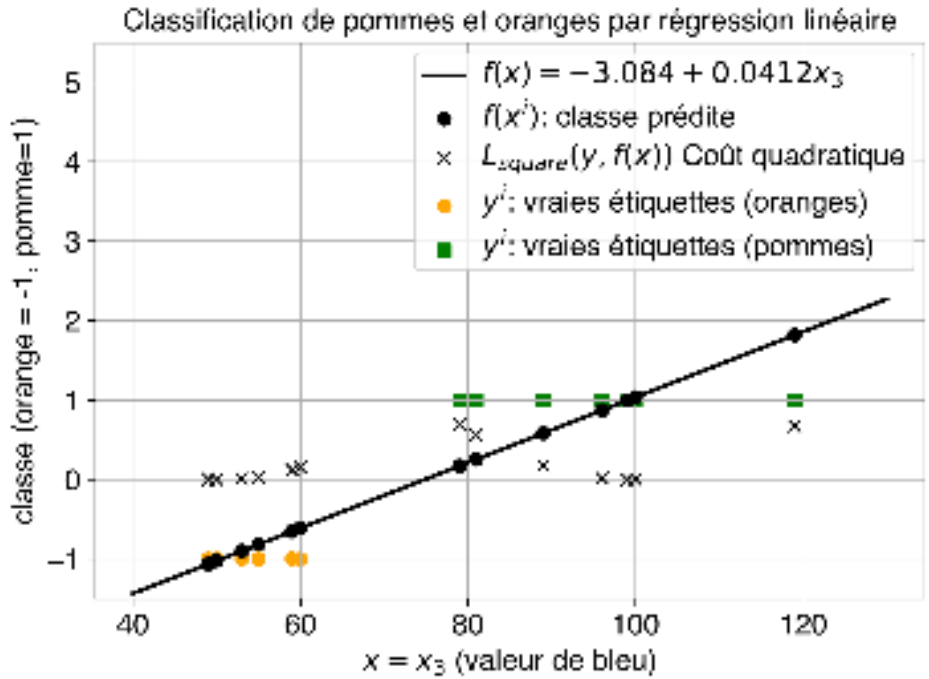


Le risque empirique avec le coût quadratique est (somme des valeurs pour les croix ($\times$) divisée par le nombre d'observations) :

$$\mathscr{R}_{14}(h) = \frac{1}{14}\sum_{i=1}^{14} L_{\text{square}}(f(\vec{x}^i), y^i) = \frac{18.316}{14}$$

Coût quadratique pour Régression linéaire pour classification binaire (solution optimale)

$$f(\vec{x}) = f(x_2) = -3.084 + 0.0412 x_2$$



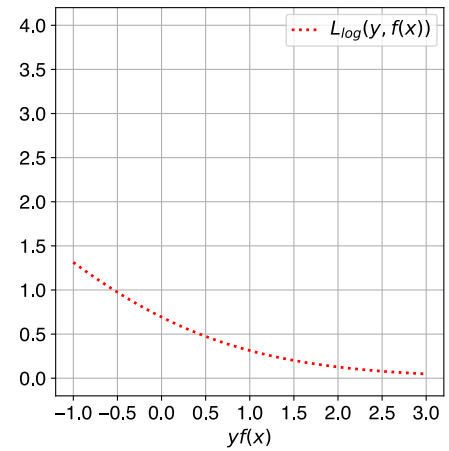
Le risque empirique avec le coût quadratique est (somme des valeurs pour les croix ($\times$) divisée par le nombre d'observations) :

$$\mathscr{R}_{14}(h) = \frac{1}{14} \sum_{i=1}^{14} L_{\text{square}}(f(\vec{x}^i), y^i) = \frac{2.55}{14}$$

Coût logistique pour la classification binaire

Définition 2.14 on appelle coût logistique (logistic loss) la fonction

$$L_{\log} : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto \log\left(1 + \exp(-yf(\vec{x}))\right)$$

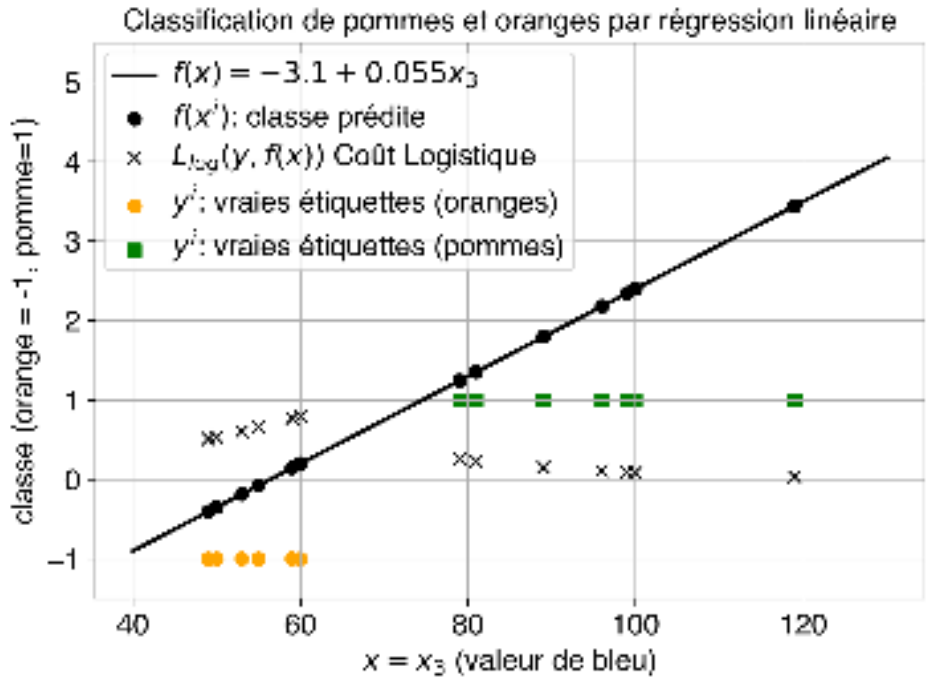


Remarques

- la valeur absolue du coût logistique quantifie notre confiance en la prédiction. On cherche alors à ce que $yf(\vec{x})$ soit la plus grande possible.
- Si on a $\mathcal{Y} = \{0, 1\}$, l'entropie croisée (voir plus loin) est équivalente au coût logistique (qui prend $\mathcal{Y} = \{-1, 1\}$)

Coût logistique pour Régression linéaire pour classification binaire

$$f(\vec{x}) = f(x_2) = -3.1 + 0.055x_2$$



Note : il y a deux points qui ont $x_2 = 59$; le risque empirique avec le coût logistique est (somme des croix $\times$ divisée par le nombre de points) :

$$\mathscr{R}_{14}(h) = \frac{1}{14}\sum_{i=1}^{14} L_{\text{log}}(f(\vec{x}^i), y^i) = \frac{5.59}{14}$$

Résumé : fonctions de perte pour la classification binaire

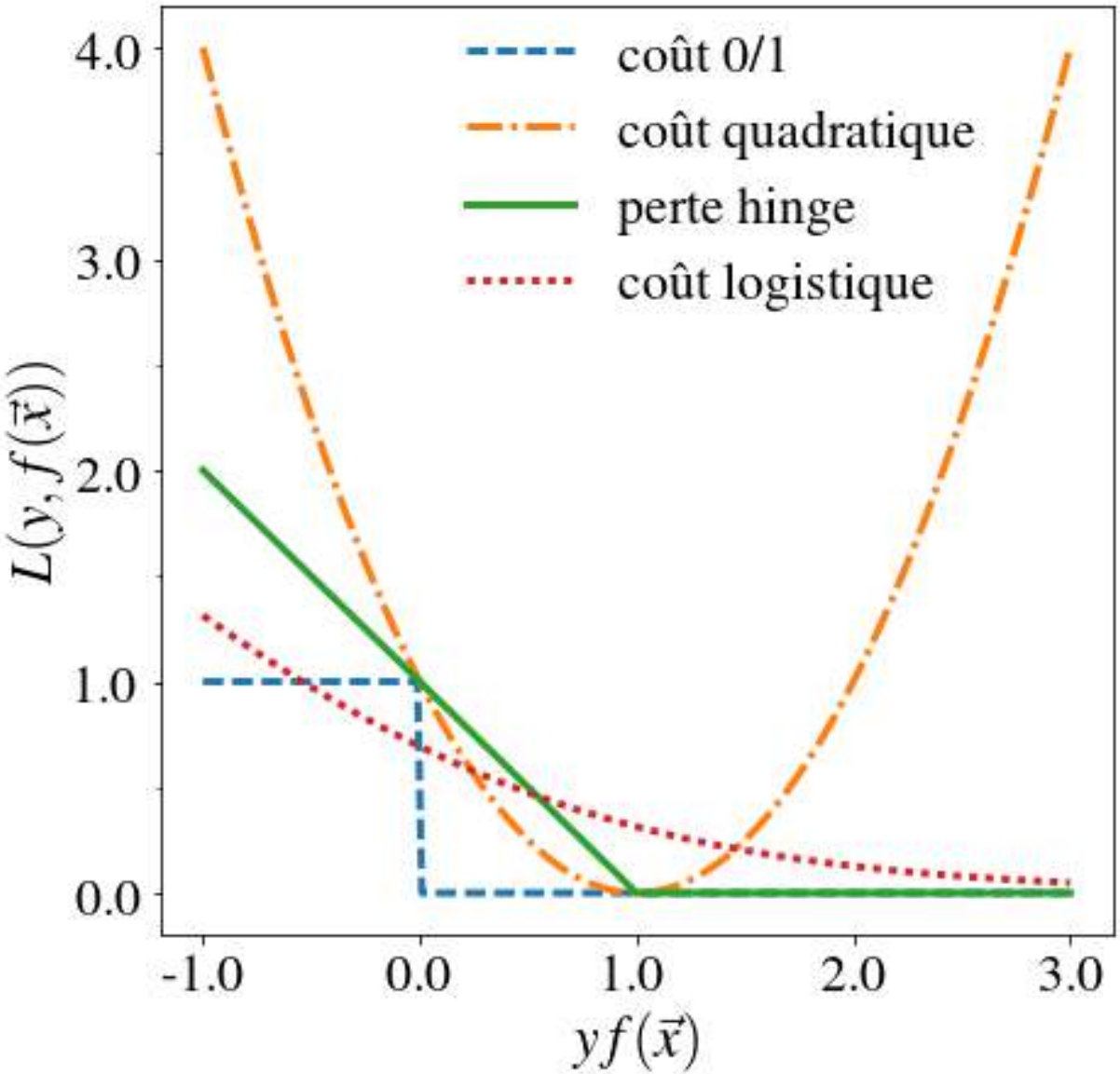


FIGURE 2.4 – Fonctions de perte pour la classification binaire.

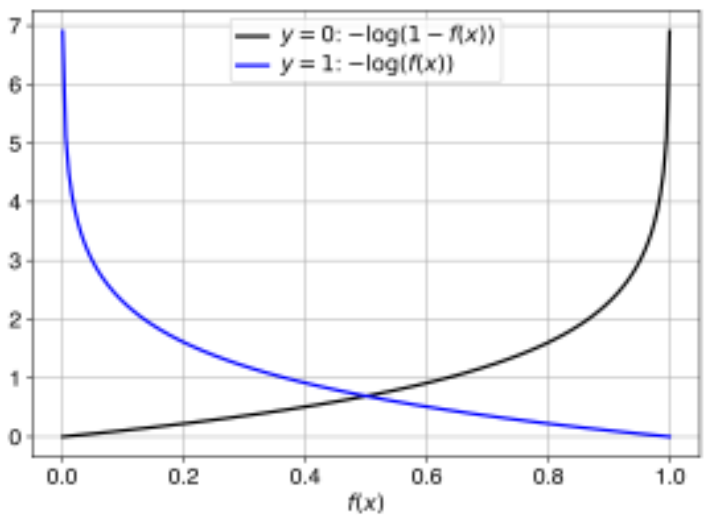
Azencott

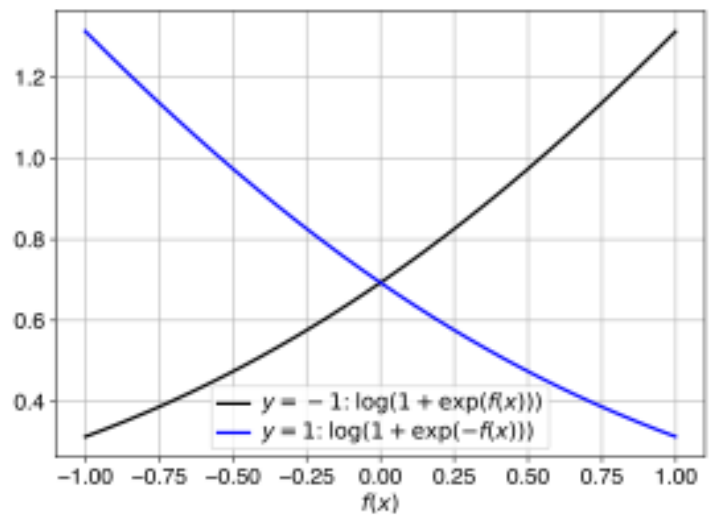
Entropie croisée pour la classification binaire

Définition 2.15 on appelle entropie croisée (cross-entropy) la fonction

$$L_H : \{0,1\} \times]0,1[\to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto -y \log f(\vec{x}) - (1-y) \log (1 - f(\vec{x}))$$
$$= \begin{cases} -\log f(1-\vec{x}) & \text{si } y = 0 \\ -\log f(\vec{x}) & \text{si } y = 1 \end{cases}$$

Note : l'entropie croisée (qui prend $\mathcal{Y} = \{0,1\}$) est équivalente au coût logistique vu précédemment (qui prenait $\mathcal{Y} = \{-1,1\}$)





Entropie croisée pour la classification multi-classe

On considère C fonctions de décision $f_c : \mathcal{X} \to \mathcal{Y}$

Définition 2.16 l'entropie croisée (cross-entropy) dans le cas multi-classe est la fonction

$$L_H : \{1, 2, \ldots, C\} \times \mathbb{R} \to \mathbb{R}$$

$$y, f(\vec{x}) \mapsto -\sum_{c=1}^{C} \delta(y, c) \log f_c(\vec{x}) = -\log f_y(\vec{x})$$

Remarques :

- On a : $\delta(y, c) = \begin{cases} 1 & \text{si } y = c \\ 0 & \text{sinon.} \end{cases}$
- $f_y(\vec{x})$ désigne la fonction de décision qui correspond à la classe véritable de $\vec{x}$, parmi les C classes possibles.

Extension de la fonction d'erreur hinge pour la classification multi-classe

On cherche à ce que la fonction de décision pour la véritable classe de $\vec{x}$, $f_y(\vec{x})$, prenne une valeur supérieure à toutes les autres fonctions de décision $f_c(\vec{x})$, pour $c \neq y$.

Plusieurs propositions :

Weston et Watkins (1999)

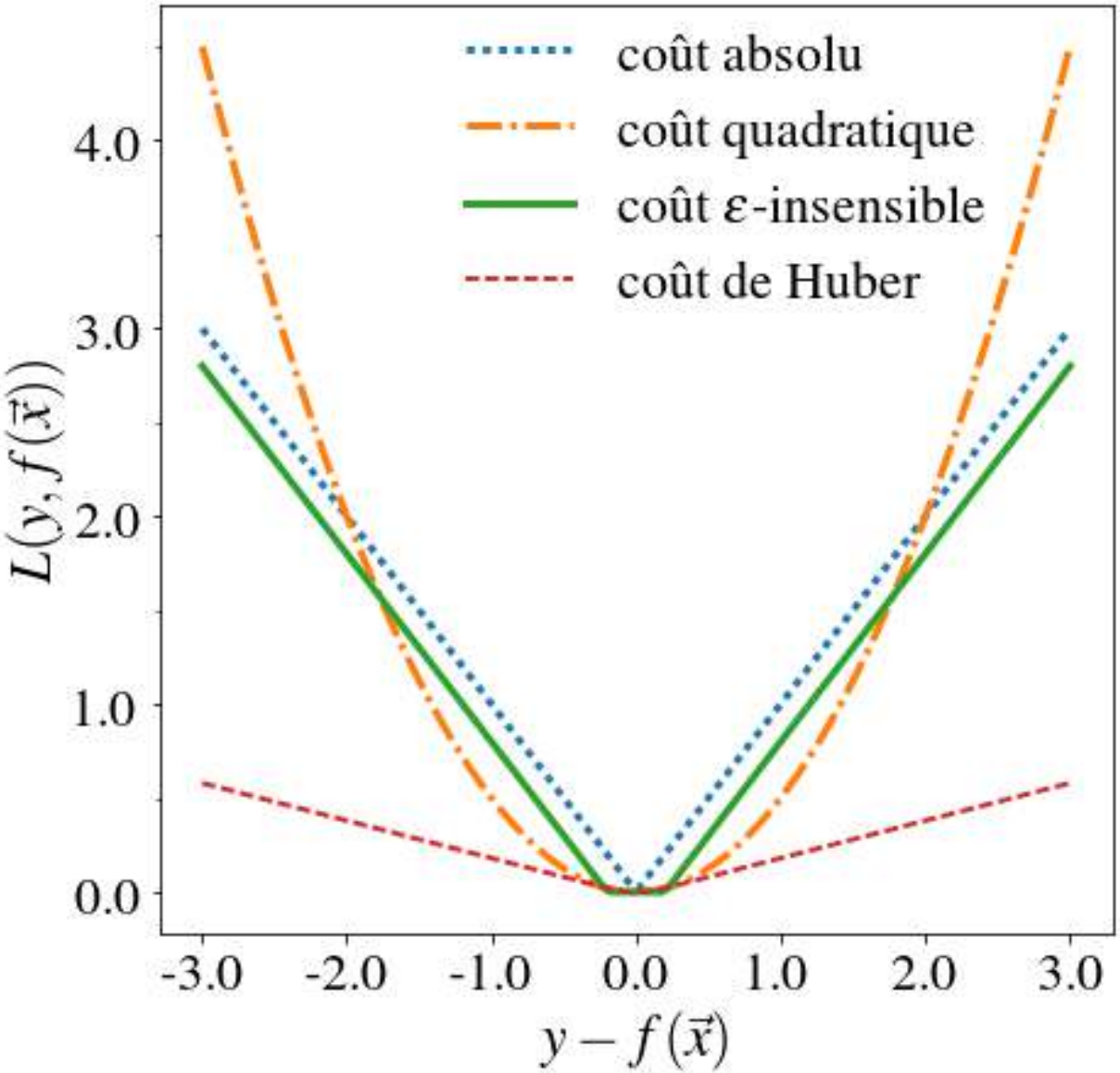
$$L_{\text{hinge}}(y, f(\vec{x})) = \sum_{c \neq y} [1 + f_c(\vec{x}) - f_y(\vec{x})]_+$$

Crammer et Singer (2001) ; utilisent un maximum à la place de la somme :

$$L_{\text{hinge}}(y, f(\vec{x})) = \left[1 + \max_{c \neq y} f_c(\vec{x}) - f_y(\vec{x})\right]_+$$

Remarque : fonctions de coût rarement utilisée (on leur préfère la perte hinge binaire dans une approche une-contre-une ou une-contre-toutes).

Fonctions de perte pour la régression



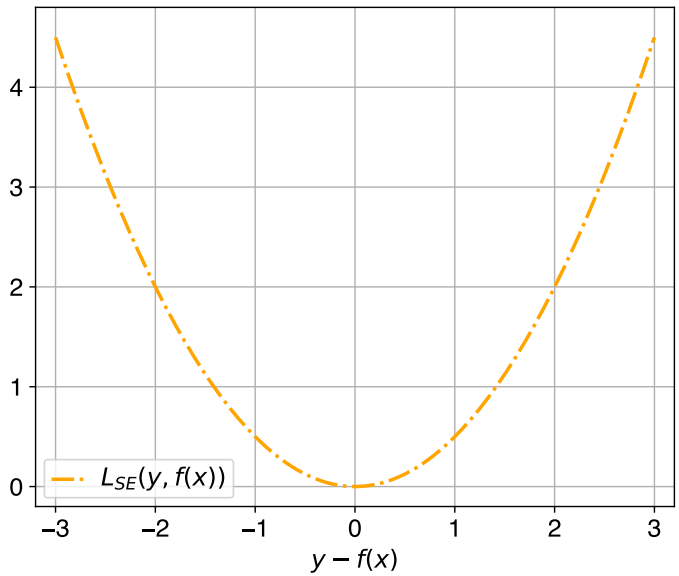
Azencott

FIGURE 2.5 – Fonctions de coût pour un problème de régression.

Coût quadratique pour la régression

Définition 2.17 on appelle fonction de coût quadratique (square loss) la fonction

$$L_{\text{SE}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto \frac{1}{2}(y - f(\vec{x}))^2$$

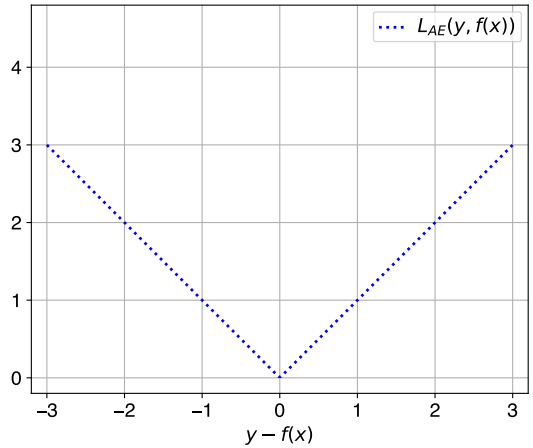


Note : le coefficient $1/2$ permet d'éviter des coefficients multiplicateurs lors de la dérivation du risque empirique pour le minimiser.

Coût absolu et coût ϵ-insensible pour la régression

Définition 2.18 on appelle fonction de coût absolu (absolute loss) la fonction

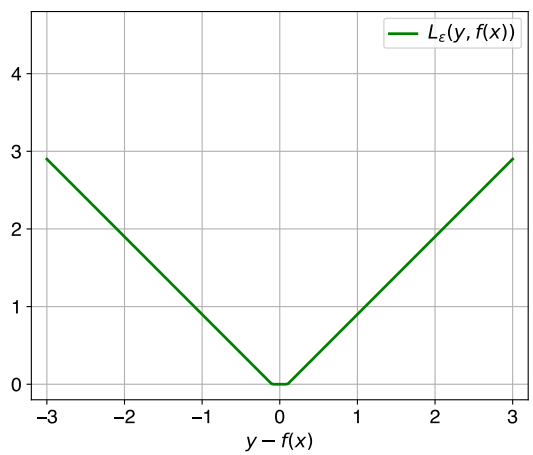
$$L_{\text{AE}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto |y - f(\vec{x})|$$



Note : même les prédictions très proches de la véritable étiquette sont pénalisées (faiblement, toutefois) $\Rightarrow$ numériquement quasi impossible d'avoir une prédiction exacte.

Définition 2.19 étant donné $\epsilon > 0$, on appelle fonction de coût ϵ-insensible (ϵ-insensitive loss) la fonction

$$L_\epsilon : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$
$$y, f(\vec{x}) \mapsto \max(0, |y - f(\vec{x})| - \epsilon).$$



note : pas dérivable en $\pm\epsilon$

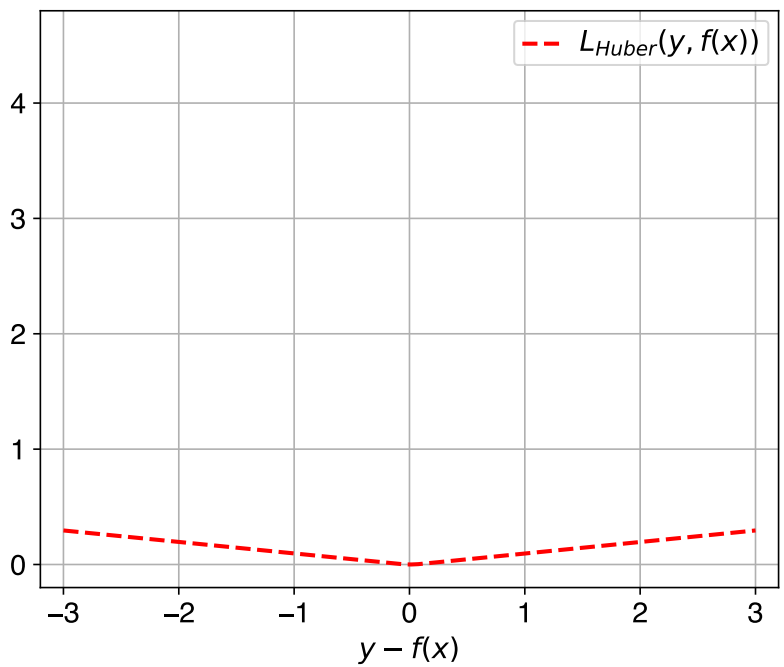
Coût de Huber pour la régression

Définition 2.20 on appelle fonction de coût de Huber (Huber loss) la fonction

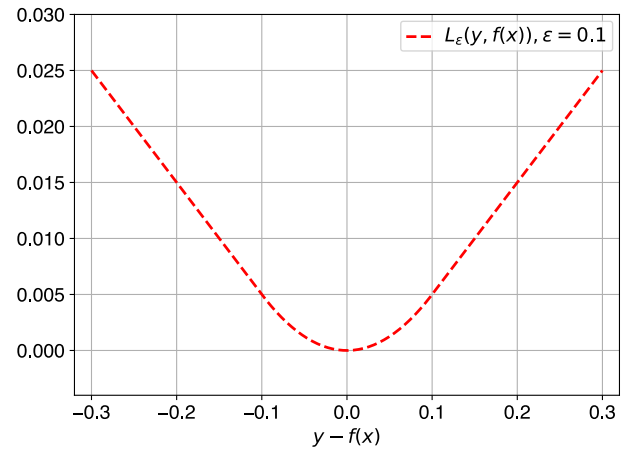
$$L_{\text{Huber}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$

$$y, f(\vec{x}) \mapsto \begin{cases} \frac{1}{2}\left(y - f(\vec{x})\right)^2 & \text{si } |y - f(\vec{x}| < \epsilon \\ \epsilon \left|y - f(\vec{x})\right| - \frac{1}{2}\epsilon^2 & \text{sinon} \end{cases}$$

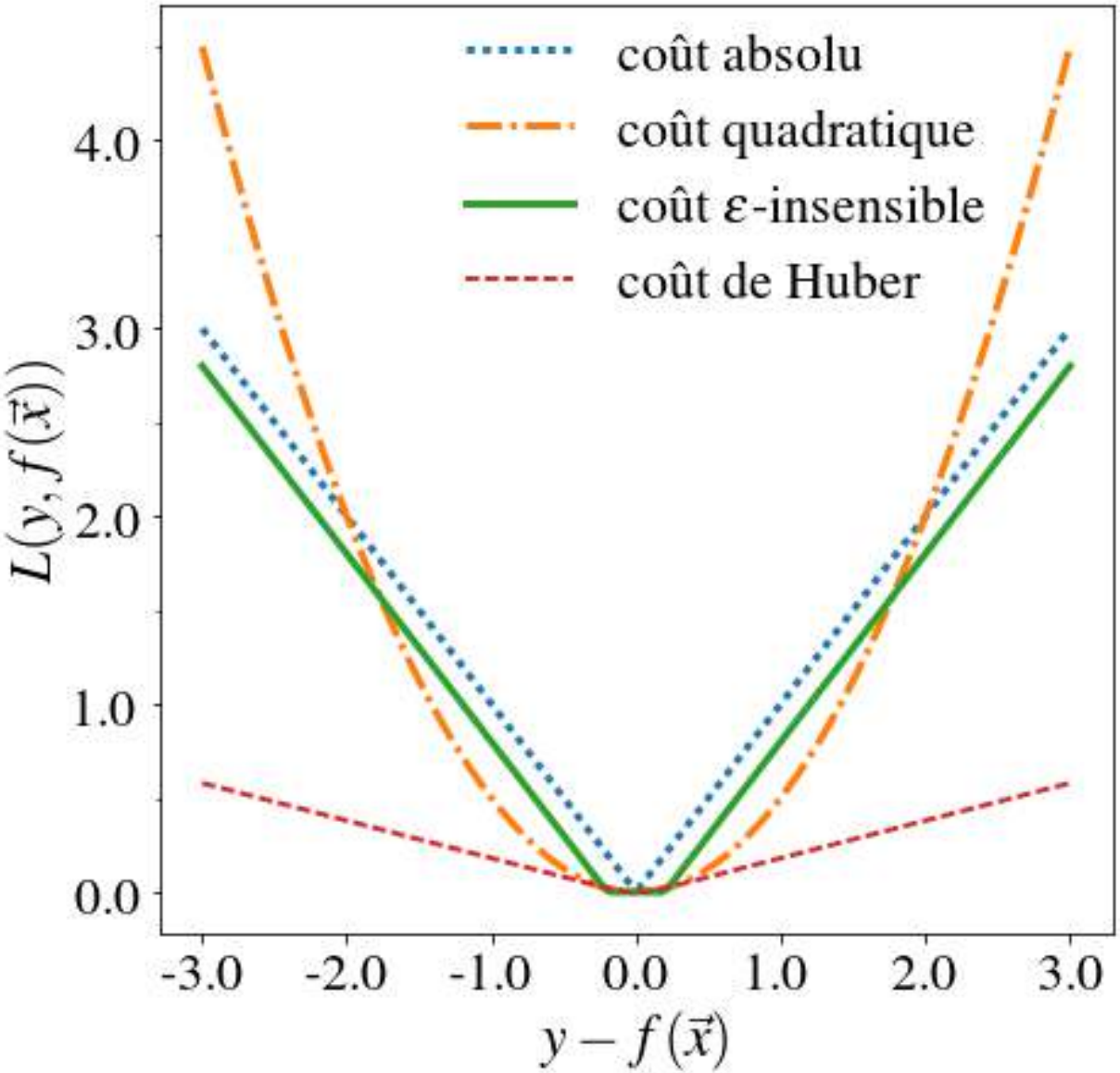
Note : Le terme $-\frac{1}{2}\epsilon^2$ permet d'assurer la continuité de la fonction.



→



Résumé : fonctions de perte pour la régression



Azencott

FIGURE 2.5 – Fonctions de coût pour un problème de régression.

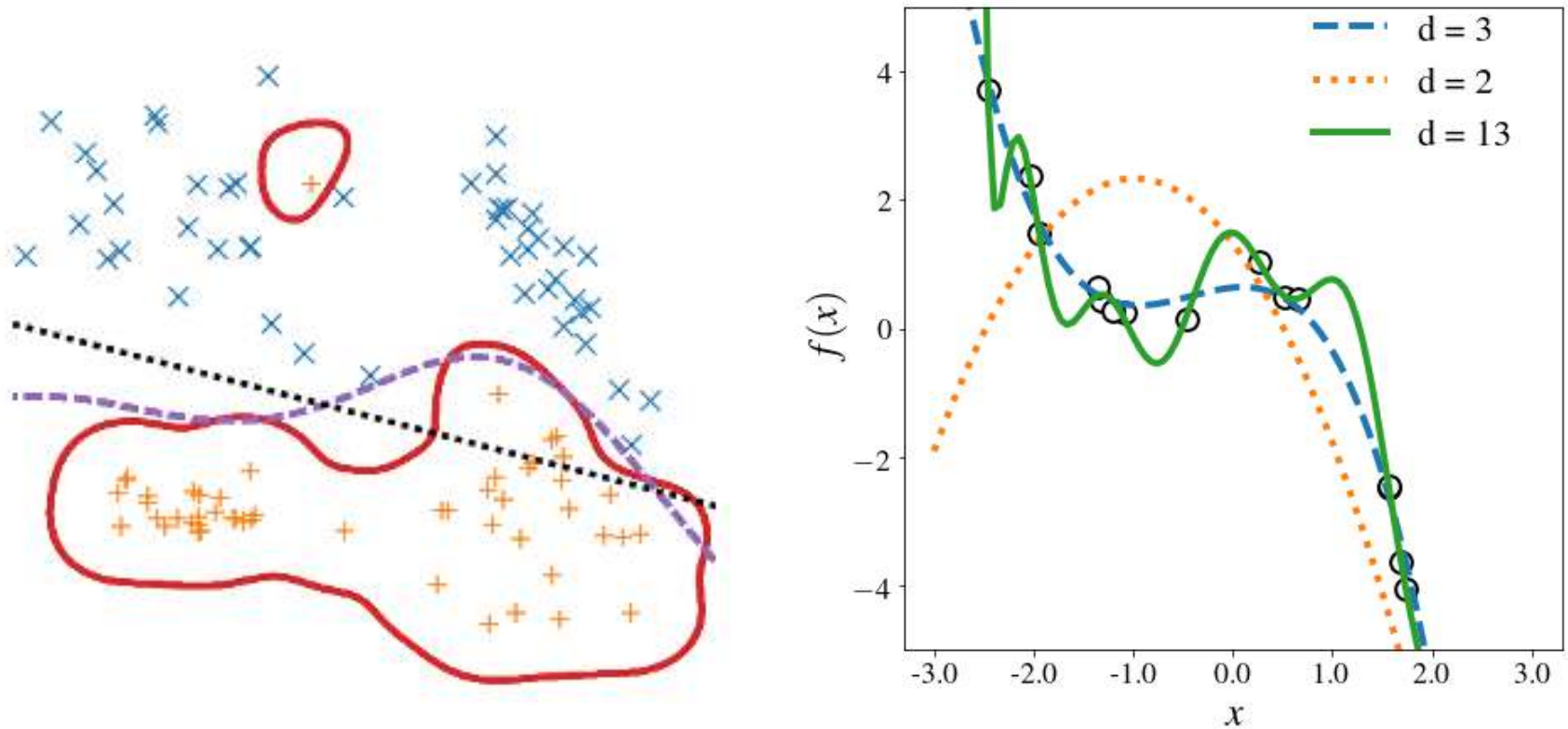
Généralisation et sur-apprentissage

Définition 2.21 (Généralisation) On appelle généralisation la capacité d'un modèle à faire des prédictions correctes sur de nouvelles données, qui n'ont pas été utilisées pour le construire.

Définition 2.22 (Sur-apprentissage) On dit d'un modèle qui, plutôt que de capturer la nature des objets à étiqueter, modélise aussi le bruit et ne sera pas en mesure de généraliser qu'il sur-apprend. En anglais, on parle d'overfitting.

Définition 2.23 (Sous-apprentissage) On dit d'un modèle qui est trop simple pour avoir de bonnes performances même sur les données utilisées pour le construire qu'il sous-apprend. En anglais, on parle d'underfitting.

Illustration : Sous-apprentissage et sur-apprentissage



(A) Pour séparer les observations négatives (x) des observations positives (+), la droite pointillée sous-apprend. La frontière de séparation en trait plein ne fait aucune erreur sur les données mais est susceptible de sur-apprendre. La frontière de séparation en trait discontinu est un bon compromis.

(B) Les étiquettes y des observations (représentées par des points) ont été générées à partir d'un polynôme de degré $d = 3$. Le modèle de degré $d = 2$ approxime très mal les données et sous-apprend, tandis que celui de degré $d = 13$, dont le risque empirique est plus faible, sur-apprend.

FIGURE 2.6 – Sous-apprentissage et sur-apprentissage

Azencott

Résumé du cours 2

- Formalisation du problème d'apprentissage supervisé : observations, étiquettes, jeu d'apprentissage
- Fonction de décision pour la classification, régions de décision, frontières de décision
- Classification multi-classe à partir de classifieurs binaires (une-contre-toutes, ou une-contre-une)
- Espace des hypothèses
- Formalisation de la tâche d'apprentissage à partir du jeu de données et d'un espace d'hypothèses : trouver une fonction de prédiction des étiquettes qui approche au mieux les données étiquetées
- Solution optimale : requiert la définition d'une fonction coût et d'une méthode d'optimisation
- Risque et risque empirique : formalise l'optimalité
- Divers coûts possibles : classification binaire (0/1), pour fonction de décision (0/1, quadratique, hinge, logistique), pour régressions (quadratique, absolu, ϵ-insensible, Huber)
- Généralisation contre sur-apprentissage : un trop grand nombre de paramètres dans le modèle pose un risque de sur-apprentissage, fit excessif

Guide de lecture pour ce cours

Chloé-Agathe Azencott “Introduction au Machine Learning”,
Dunod, 2019, ISBN 978-210-080153-4
Chapitre 2 : Apprentissage supervisé