Deep learning par la pratique

Leçon 1 : Sur et sous entraînement

Présenté par **Morgan Gautherot**

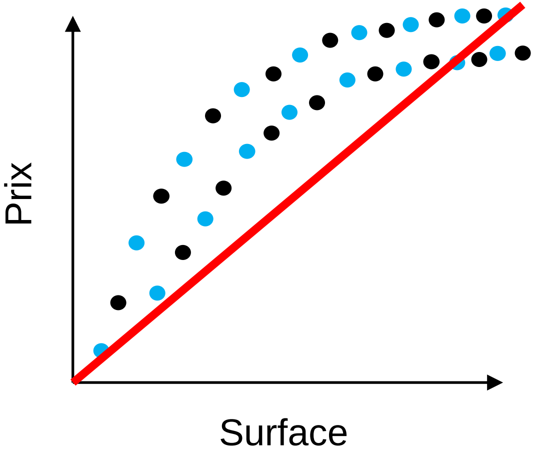
Train / dev / test sets

Sur-entraînement et sous-entraînement

Sous-entraînement

Erreur sur le jeu d’entraînement : Élevée

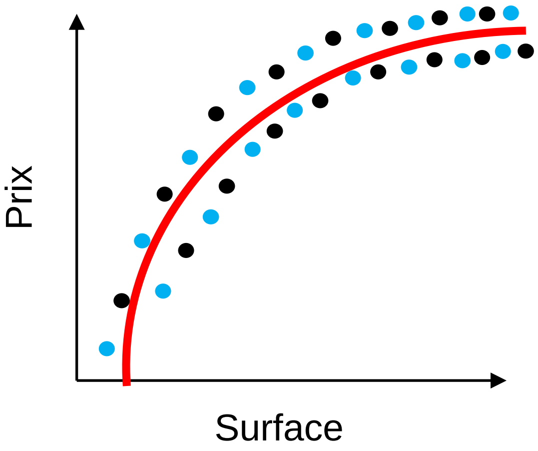
Erreur sur le jeu de test : Élevée



Entraînement correct

Erreur sur le jeu d’entraînement : Faible

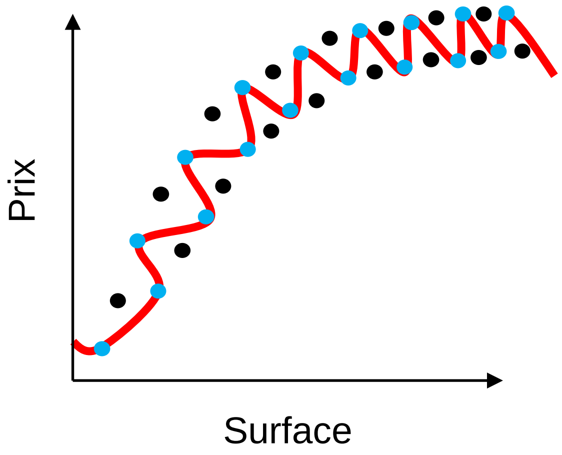
Erreur sur le jeu de test : Faible



Sur-entraînement

Erreur sur le jeu d’entraînement : Nulle

Erreur sur le jeu de test : Moyenne



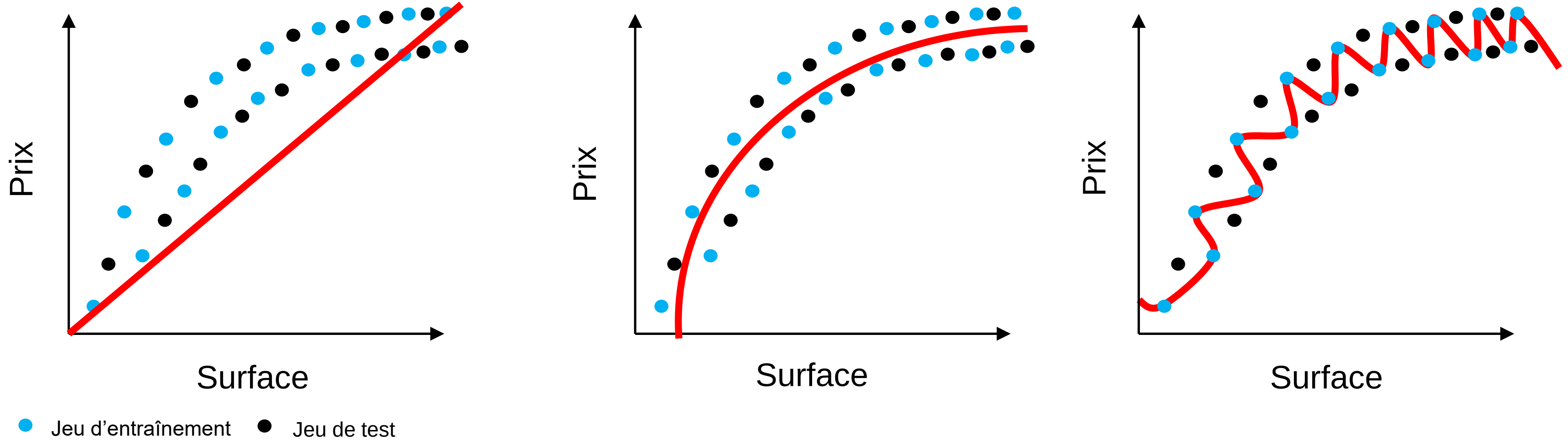
● Jeu d’entraînement ● Jeu de test

Complexité du modèle

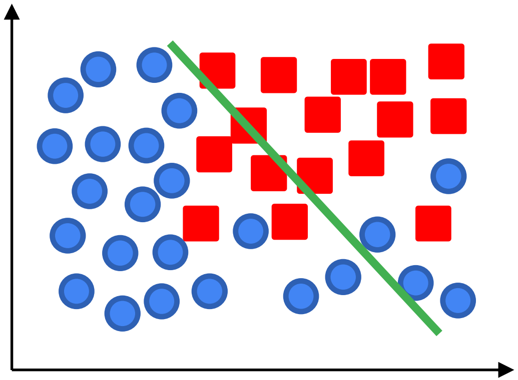
$$\hat{y} = w_0 + w_1.x_1$$

$$\hat{y} = w_0 + w_1.x_1 + w_2.{x_1}^2$$

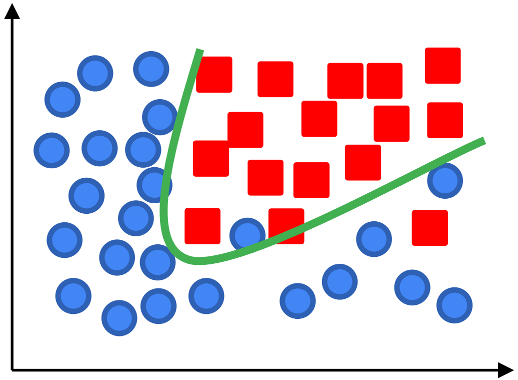
$$\hat{y} = w_0 + w_1.x_1 + w_2.{x_1}^2 + w_3.{x_1}^3 + w_4.{x_1}^4 + \cdots$$



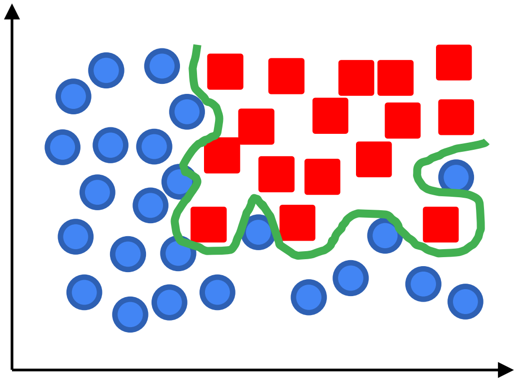
Pour la classification



Sous-entraînement



Entraînement correct



Sur-entraînement

Deep learning par la pratique

Leçon 2 : La pénalisation L2

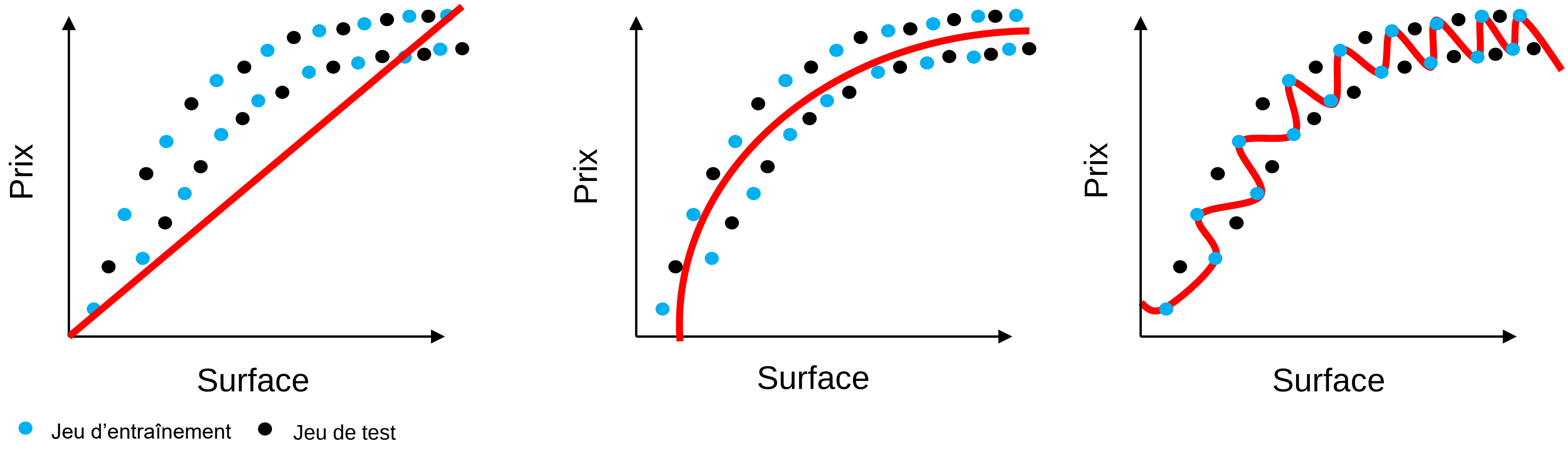
Présenté par **Morgan Gautherot**

Complexité du modèle

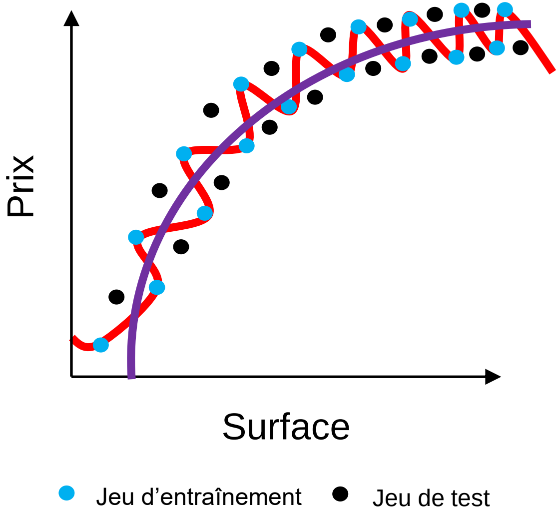
$$\hat{y} = w_0 + w_1 . x_1$$

$$\hat{y} = w_0 + w_1 . x_1 + w_2 . {x_1}^2$$

$$\hat{y} = w_0 + w_1 . x_1 + w_2 . {x_1}^2 + w_3 . {x_1}^3 + w_4 . {x_1}^4 + \cdots$$



Pénalisation des paramètres



$$\hat{y} = w_0 + w_1.x_1 + w_2.x_1{}^2 + w_3.x_1{}^3 + w_4.x_1{}^4 + \cdots$$

Pénalisation des paramètres

$$\min_{w} J(w) = \underbrace{\frac{1}{2m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2}_{\text{Minimiser l'erreur de prédiction}} \underbrace{+1000.w_3 + 1000.w_4 + \cdots}_{\text{Minimiser la valeur des paramètres } w_3, w_4, \ldots}$$

Régression Ridge ou pénalisation L2

- Un modèle avec des paramètres plus homogène est moins sujet au sur-entraînement.

Paramètre de régularisation

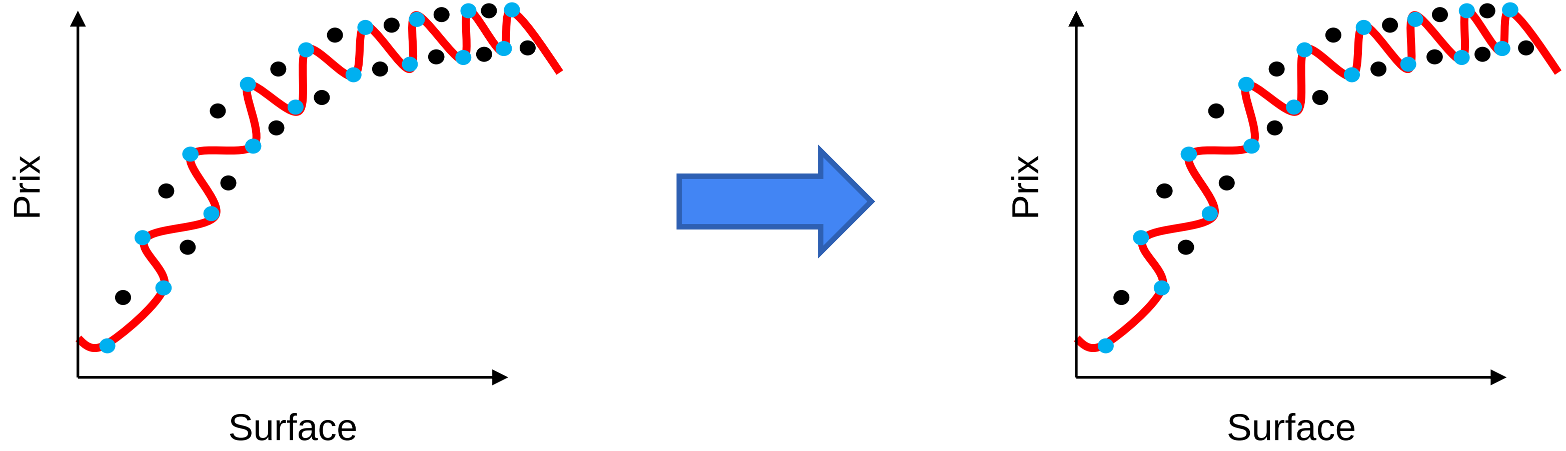
$$J(w) = \frac{1}{2m}\left[\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2 + \boxed{\lambda \sum_{j=1}^{n} w_j^2}\right]$$

Régularisation

Impact du coefficient de régularisation

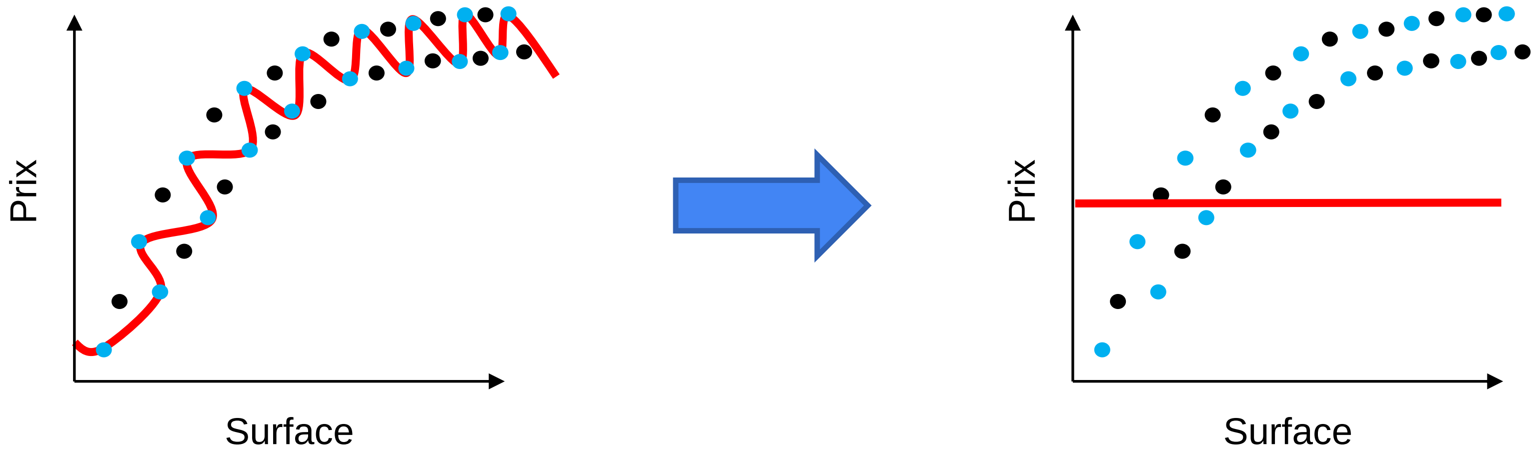
λ trop petit

$$J(w) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(\hat{y}^{(i)} - y^{(i)}\right)^2 \cancel{+ \lambda \sum_{j=1}^{n} w_j^{\,2}}\right]$$



Impact du coefficient de régularisation

λ trop grand $\quad J(w) = \dfrac{1}{2m}\left[\cancel{\sum_{i=1}^{m}\left(\hat{y}^{(i)} - y^{(i)}\right)^2} + \lambda \sum_{j=1}^{n} {w_j}^2\right]$

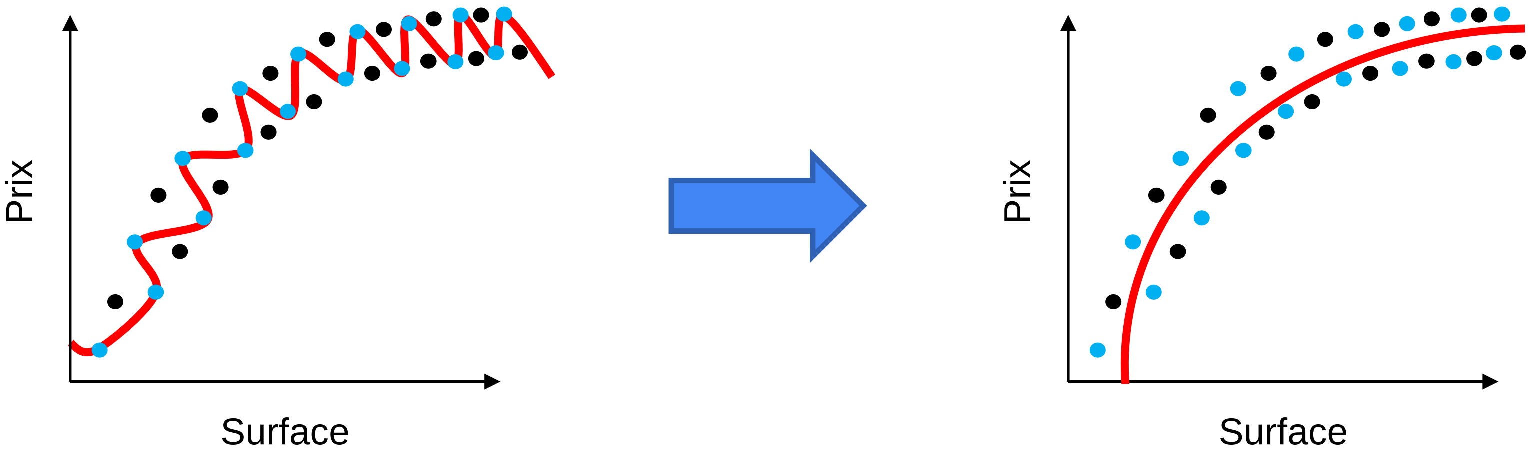


Impact du coefficient de régularisation

[0.01, …, 0.1, …., 0.5]

λ adéquat

$$J(w) = \frac{1}{2m}\left[\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2 + \lambda\sum_{j=1}^{n} w_J{}^2\right]$$



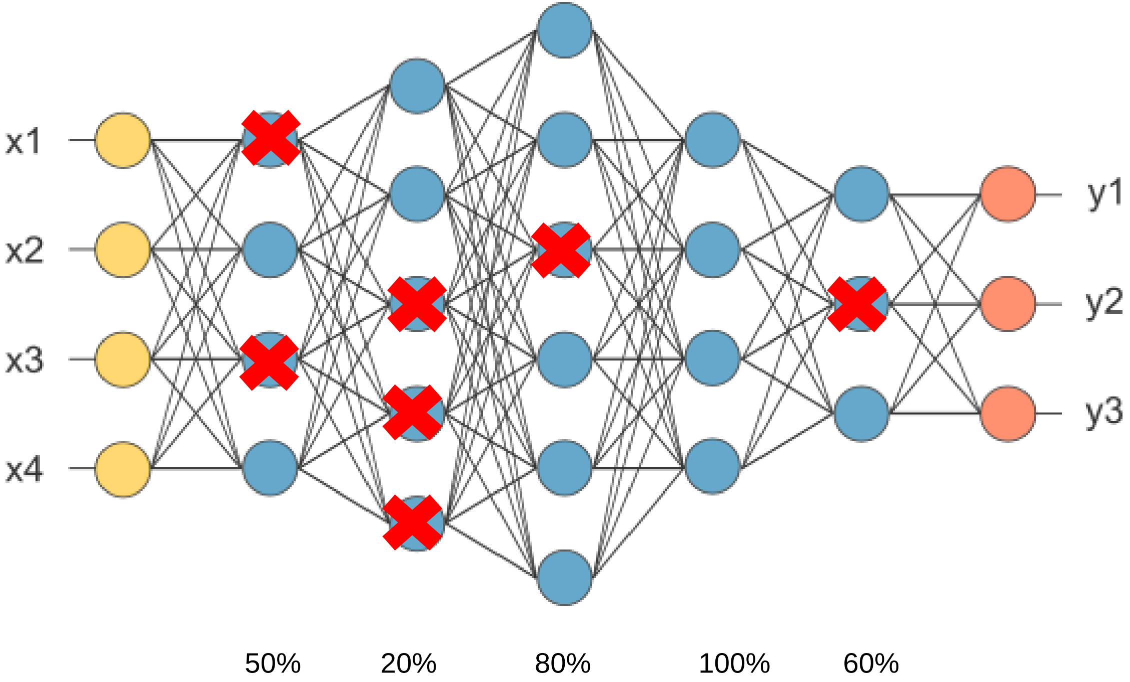
Deep learning par la pratique

Leçon 3 : Le drop out

Présenté par **Morgan Gautherot**

Le drop out

En utilisant le drop-out, vous ne pouvez pas compter sur une seule fonction, vous devez donc répartir les poids.



Deep learning par la pratique

Leçon 4 : La data augmentation

Présenté par **Morgan Gautherot**

La data augmentation

(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

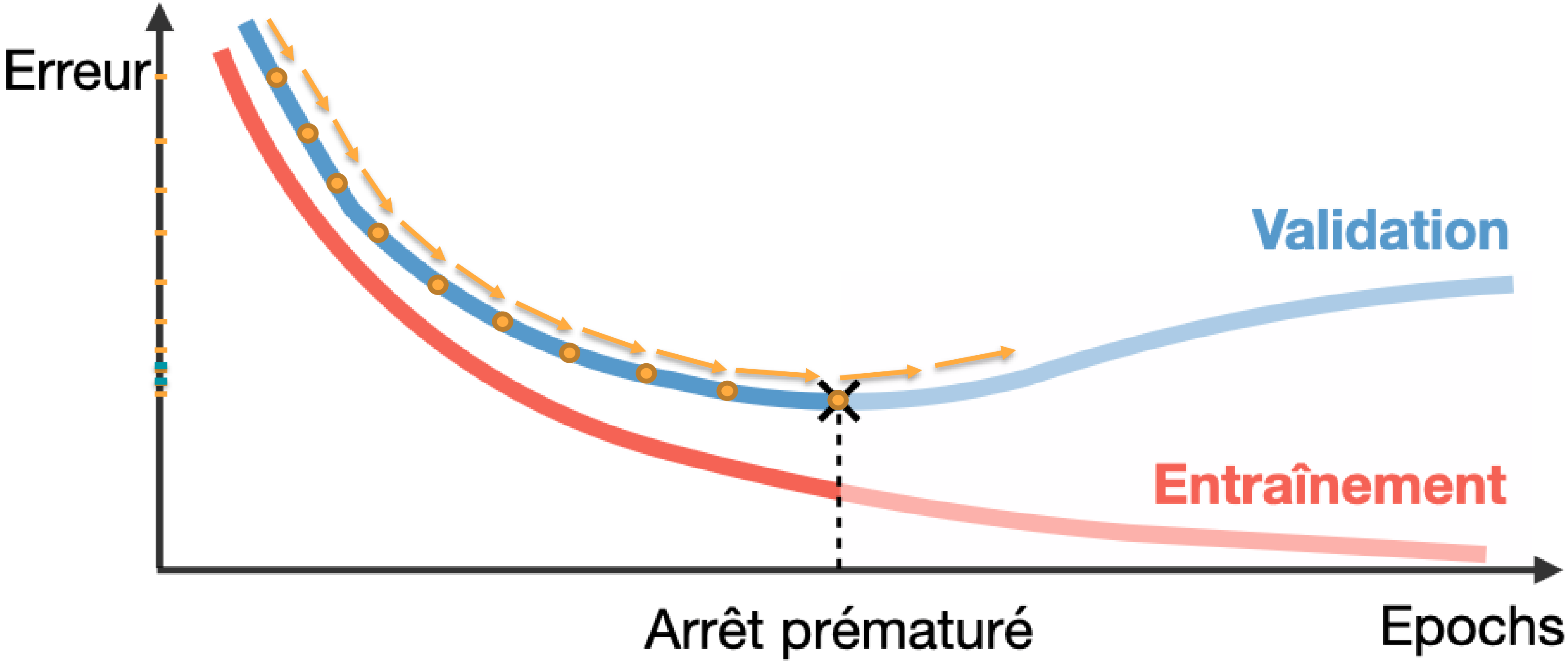
(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Deep learning par la pratique

Leçon 5 : L’early stopping

Présenté par **Morgan Gautherot**

L’early stopping

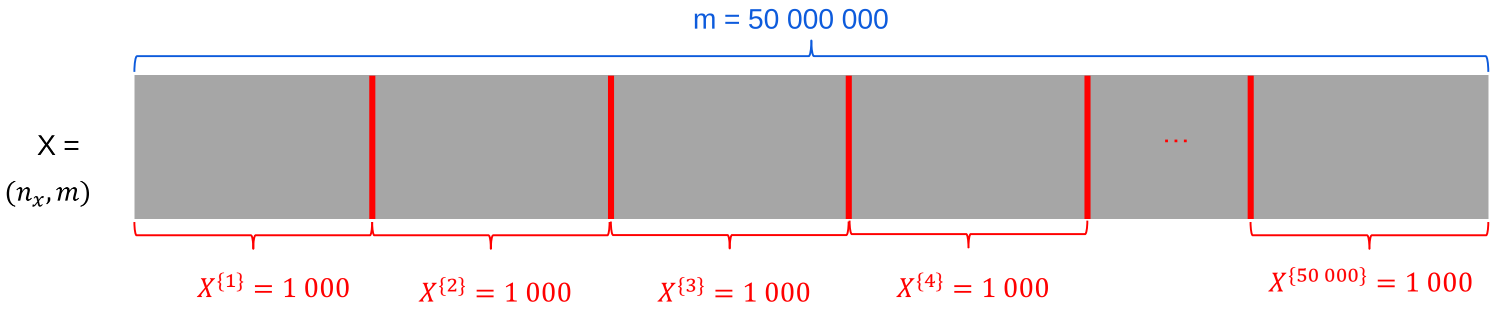


Deep learning par la pratique

Leçon 6 : Mini batch

Présenté par **Morgan Gautherot**

Batch vs mini-batch gradient descent



For epoch = 1, …, 1 000

For I = 1, …, 50 000

Forward prop $X^{\{i\}}$

Calcul du coût $X^{\{i\}}$

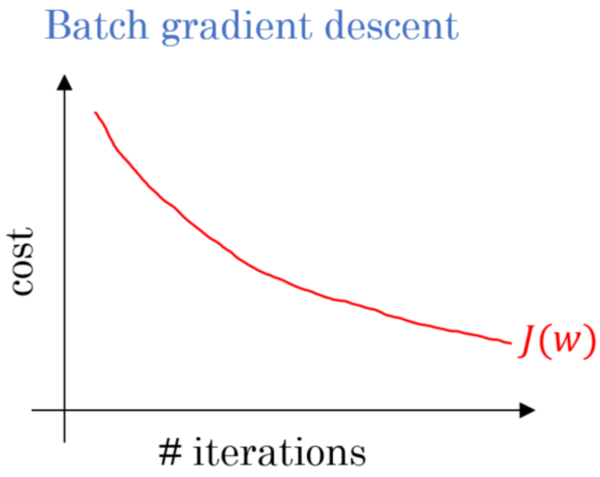
Back prop $X^{\{i\}}$

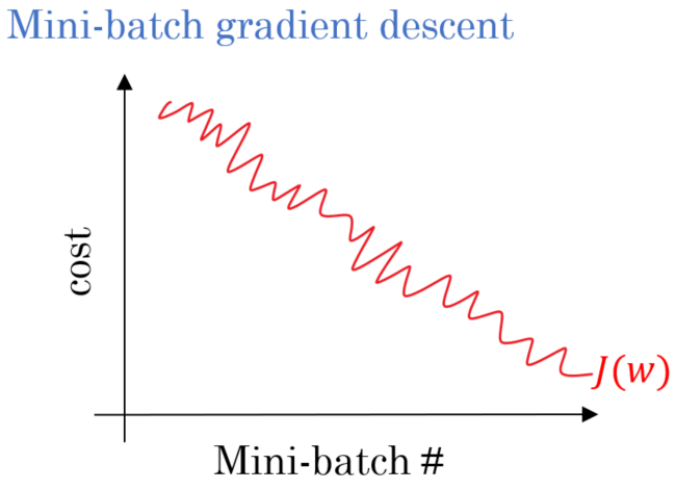
Mise à jour de w et b

1 mini batch

1 epoch

Batch vs mini-batch gradient descent





Le batch est trop grand :

Trop long par itération

Le batch est trop petit :

Perte de la vectorisation

Utilisez une puissance de 2 et assurez-vous que votre mini batch correspond à la mémoire de votre GPU.

Deep learning par la pratique

Leçon 7 : Batch normalisation

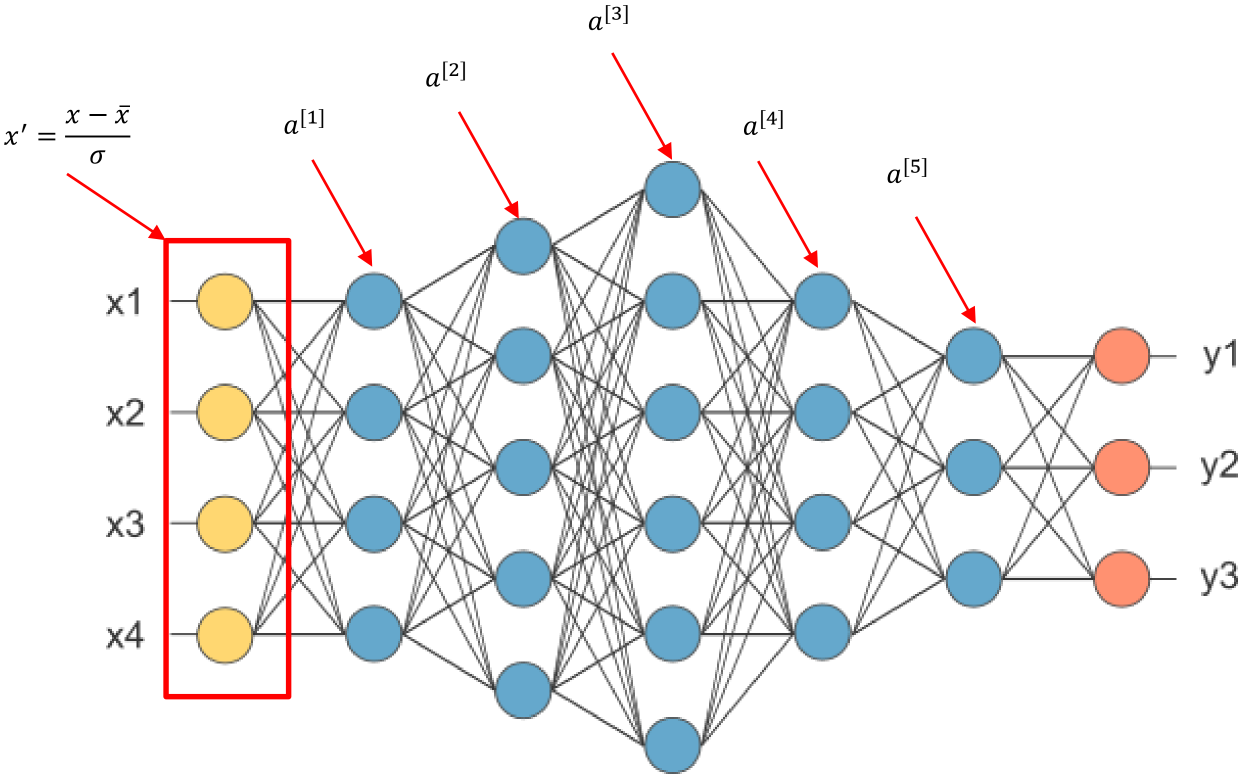
Présenté par **Morgan Gautherot**

Normalisation des données d’entrées

$$x' = \frac{x - \bar{x}}{\sigma}$$

Calculer $\bar{x}$ et σ avec votre ensemble d'entraînement et sauvegardez-les pour les appliquer à l'ensemble de dev et de test.

Modèle dense



Disparition et explosion de gradient

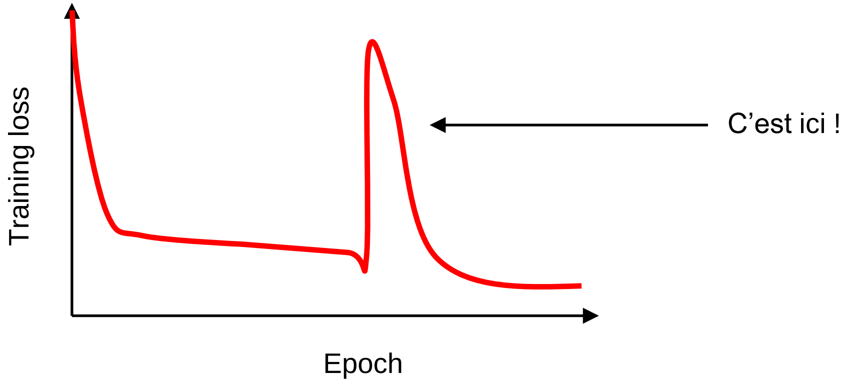
- Lorsque vous construisez un réseau neuronal très profond, votre gradient dans la dernière couche cachée peut être très important ou proche de zéro.

- Il s'agit d'un problème énorme car, avec ce problème, votre modèle ne peut pas apprendre correctement.

Les explosions de gradient

Les explosions de gradients sont faciles à détecter

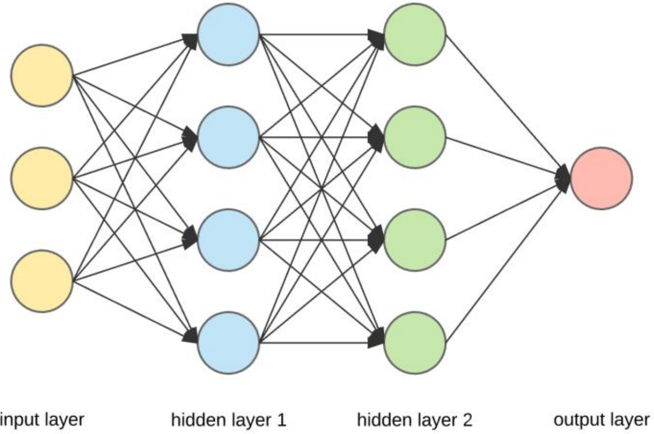
Courbe d'apprentissage instable



Les gradients peuvent être trop grands et contenir des NaNs.
et vous vous retrouvez avec des NaN dans les poids.

Batch normalization

Couche de
batch normalisation



Pour chaque couche

$$\mu^l = \frac{1}{m}\sum_i {z_i}^l$$

$${\sigma^l}^2 = \frac{1}{m}\sum_i ({z_i}^l - \mu^l)^2$$

$${Z_{norm}}^l = \frac{z^l - \mu^l}{\sqrt{{\sigma^l}^2 - \varepsilon}}$$

$$\tilde{Z}^l = \gamma\, {Z_{norm}}^l + \beta$$

$$X \xrightarrow{(w^{[1]}, b^{[1]})} z^{[1]} \xrightarrow{(\gamma^{[1]}, \beta^{[1]})} \tilde{z}^{[1]} \xrightarrow{g(\tilde{z}^{[1]})} a^{[1]}$$

$$a^{[1]} \xrightarrow{(w^{[2]}, b^{[2]})} z^{[2]} \xrightarrow{(\gamma^{[2]}, \beta^{[2]})} \tilde{z}^{[2]} \xrightarrow{g(\tilde{z}^{[2]})} a^{[2]}$$

$$a^{[2]} \xrightarrow{(w^{[3]}, b^{[3]})} z^{[3]} \xrightarrow{(\gamma^{[3]}, \beta^{[3]})} \tilde{z}^{[3]} \xrightarrow{\sigma(\tilde{z}^{[3]})} a^{[3]} \longrightarrow \hat{y}$$

Deep learning par la pratique

Leçon 8 : L’entraînement, un processus itératif

Présenté par **Morgan Gautherot**

Comment créer son premier modèle ?

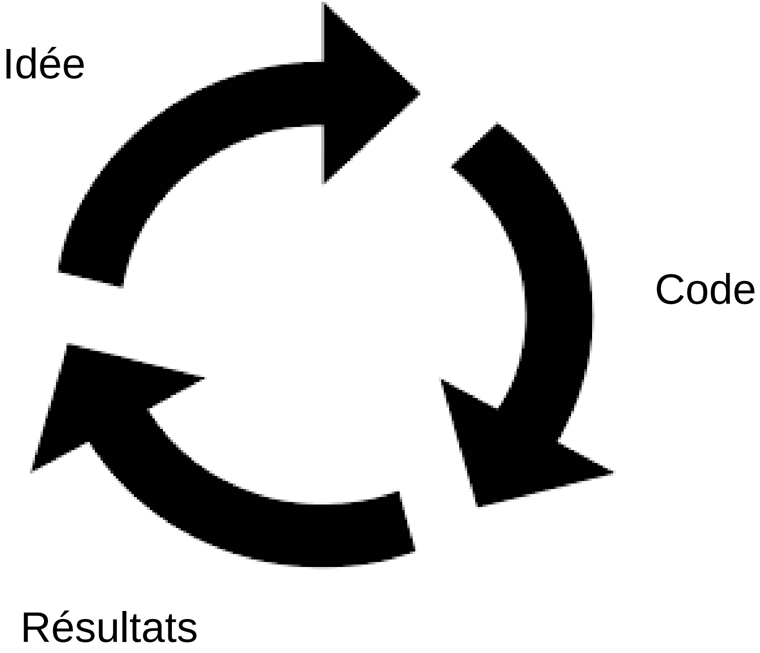
Nombre de couches

Nombre de neurones

Learning rates

Fonctions d’activations

…



Commencer simple

- Tester le modèle le plus simple possible
- Complexifier pour chercher le sur-apprentissage
- Appliquer une régularisation pour chercher la complexité idéale

Tuning process

- $\boxed{\alpha}$
- Nombre de couches
- Nombre de neurones
- Taille du Mini-batch