Linear Regression

By Dr. Nzamba Bignoumba

$y=-ax+b$

Average training duration: 4 hours 00 minute

Outline

Machine learning overview

IA

Que pouvons nous faire avec ?

Natural Language Processing
Computer Vision
Signal Processing
⋮

Machine learning overview

Synthèse du contenu d'un ou plusieurs documents Traduction d'une langue à une autre Génération du code ⋮	OpenAI-ChatGPT DeepSeek GitHub Copilot Cursor | Codex

Natural Language Processing
Computer Vision
Signal Processing
⋮

Génération d'images & d'illustrations Classification d'images médicales Classification d'images agricoles Génération de vidéos ⋮	Midjourney Canva Aidoc IA Agri AgriHyphen AI

Prévisions météorologiques Prévisions de maladies et de mortalité Prévisions boursières Prévisions de consommation d'électricité Détection d'anomalies (cybersécurité) ⋮	AWS SageMaker – DeepAR Nixtla-TimeGPT Meta-Prophet Zindi Africa Amini

Machine learning overview

IA

Que pouvons nous faire avec ?

Natural Language Processing Computer Vision Signal Processing
⋮

Avec quels types d'algorithmes?

Machine Learning

Deep Learning

Machine learning overview

K-NN

Logistic Regression

K-Means

Support Vector Machines

Decision Trees

Gradient Boosting Machines

Random Forest

Linear Regression

Principal Component Analysis

...

Machine Learning

Deep Learning

Generative Adversarial Neural Network

Varational Autoencoder

Feed Forward Neural Network

Autoencoder

State Space Model

Word Embeddings

Graph Neural Network

Neural Ordinary Differential Equations

Diffusion Model

Recurrent Neural Network

Neural Radiance Field

Transformer

Normalizing Flows

...

Machine learning overview

IA

Que pouvons nous faire avec ?

Natural Language Processing Computer Vision Signal Processing
⋮

Avec quels types d'algorithmes?

Machine Learning

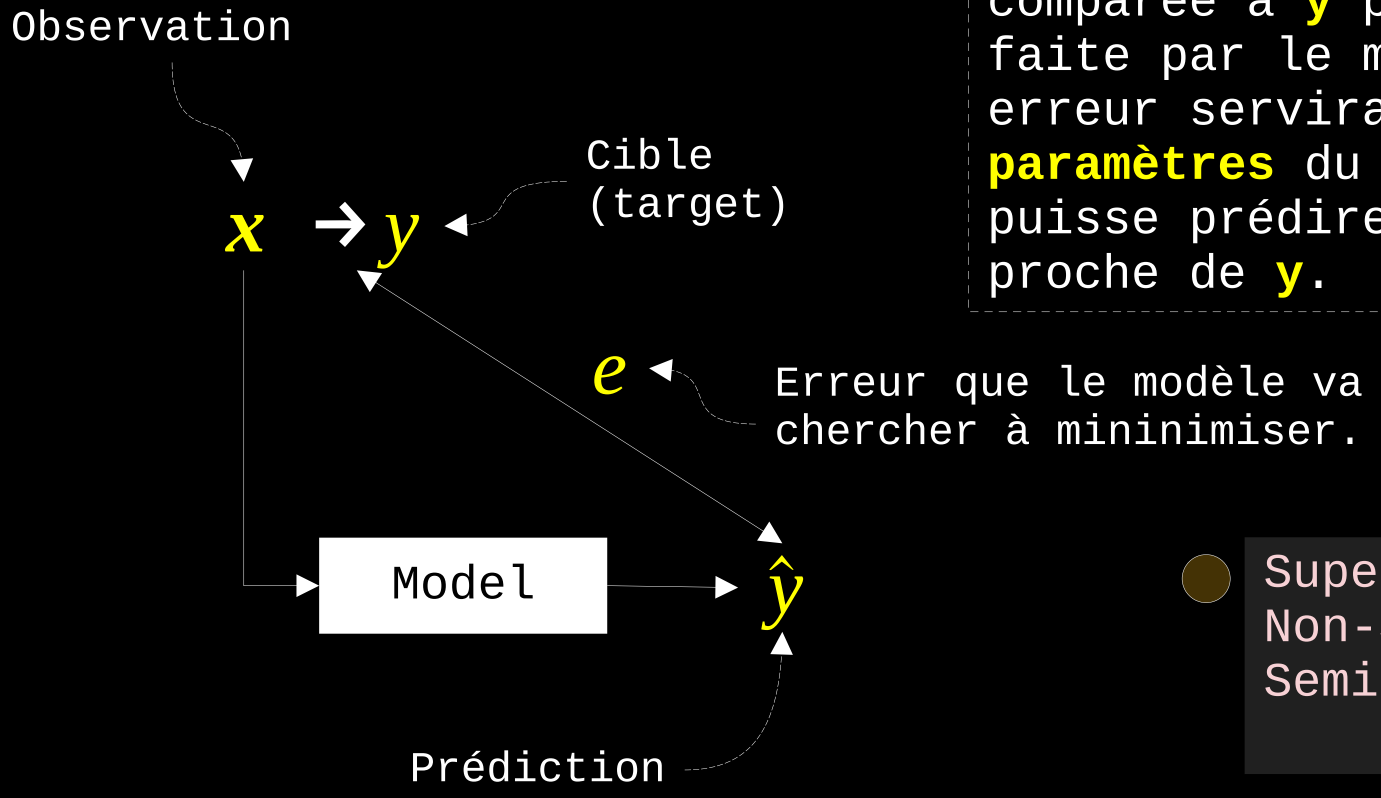
Deep Learning

Comment fonctionnent-ils ?

Supervised
Non-Supervised
Semi-Supervised

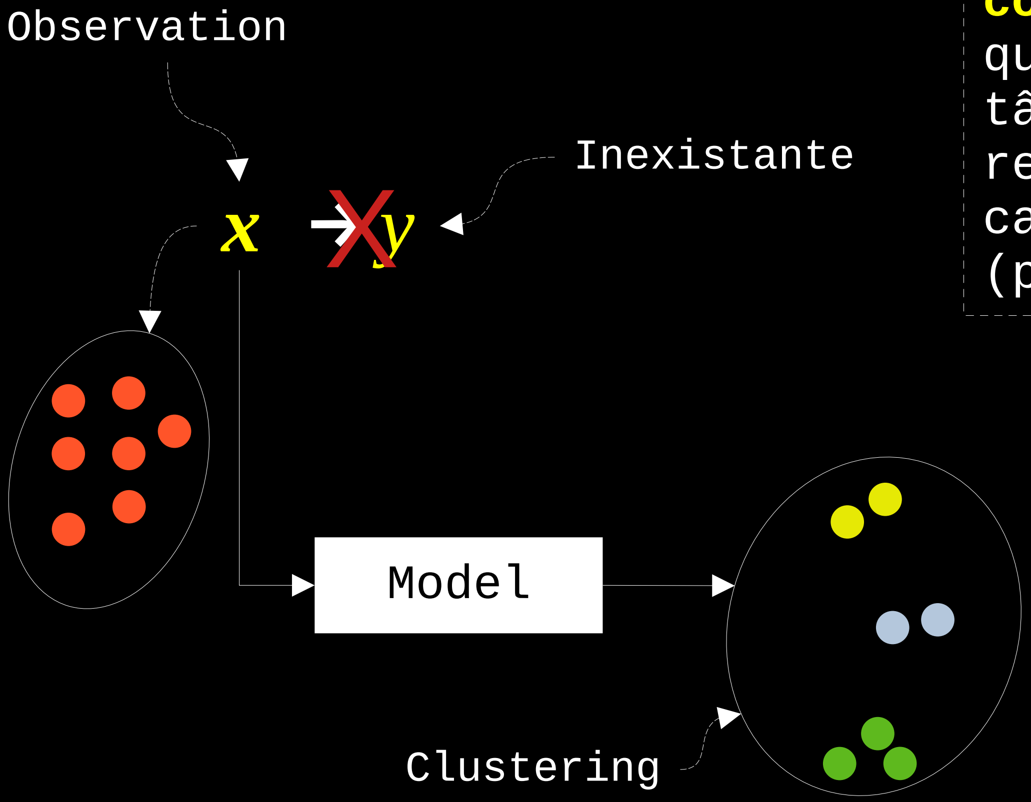
Machine learning overview

Nous disposons des données **Xs** et des des cibles correspondantes **ys**. Le modèle utilisera **X** pour prédire une valeur **ŷ**, **modèle(X)=ŷ**. **ŷ** sera ensuite comparée à **y** pour calculer l’erreur **e** faite par le modèle, **|y-ŷ|=e**. Cette erreur servira à **ajuster les paramètres** du modèle afin qu'il puisse prédire une valeur **ŷ** très proche de **y**.



- Supervised
- Non-Supervised
- Semi-Supervised

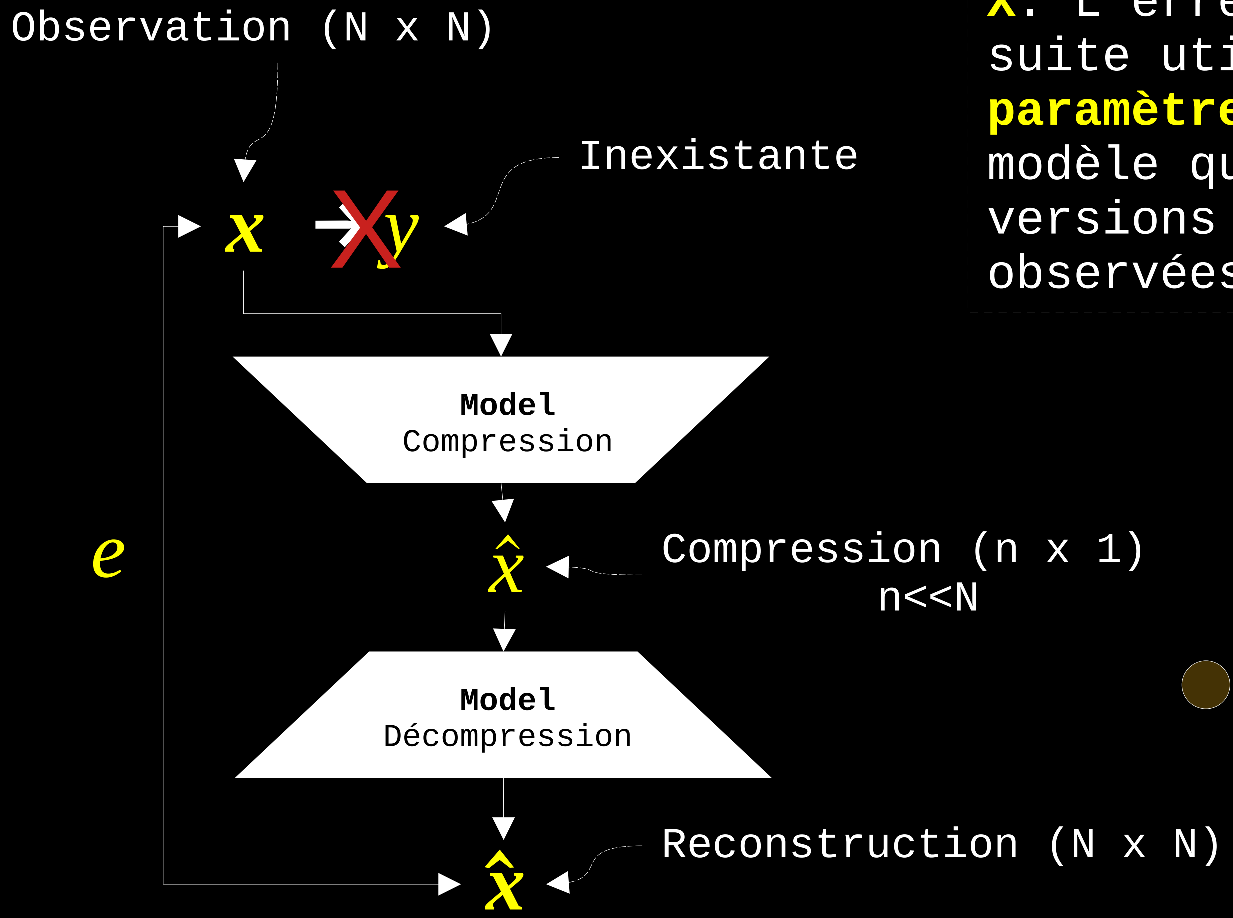
Machine learning overview



Nous ne disposons que des données **Xs**. Aucune cible **y** correspondante n'est disponible. Le modèle exploitera les **similarités** des données et leur **cooccurrence** pour effectuer la tâche qui lui est assignée. Par exemple, la tâche de clustering, qui consiste à regrouper des données présentant des caractéristiques similaires (patterns).

Supervised
Non-Supervised
Semi-Supervised

Machine learning overview



Nous ne disposons que des données **Xs**. Le modèle prend une observation **X** en entréé, **model(X)=X̂,** et compare sa prédiction **X̂** à cette même obervation **X**. L'erreur **|X-X̂|=e**, sera par la suite utilisée pour **ajuster les paramètres** du modèle. Exemple : un modèle qui apprend à créer des versions compressée des données observées.

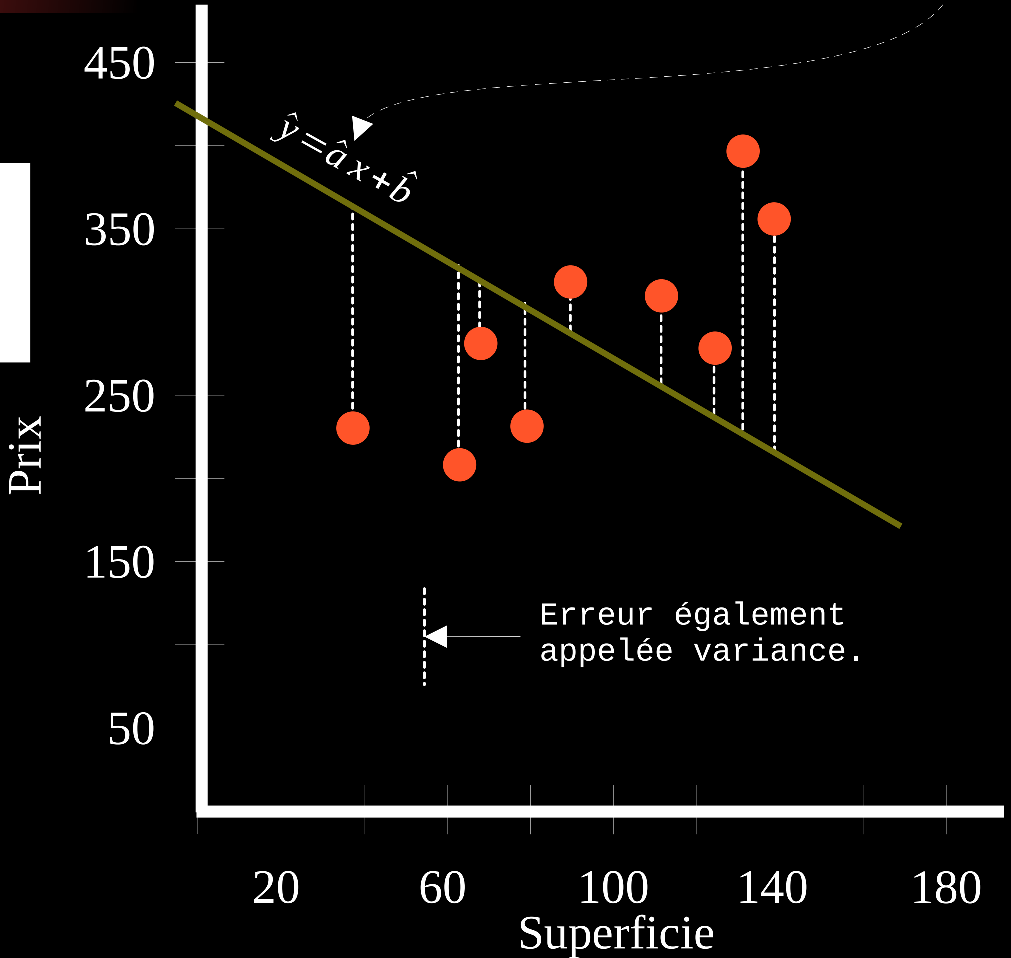
Supervised
Non-Supervised
Semi-Supervised

Theory

Model→ Linear Regression= $\hat{y}=\hat{a}x+\hat{b}$

$\hat{a}<0, \hat{b}\simeq 425$

Graphique des prix (en XAF) de location des maisons en fonction de leur superficie (en m²).



L'entraînement consiste à trouver les valeurs **â** et **b̂** qui minimisent au maximum l'erreur (**E**).

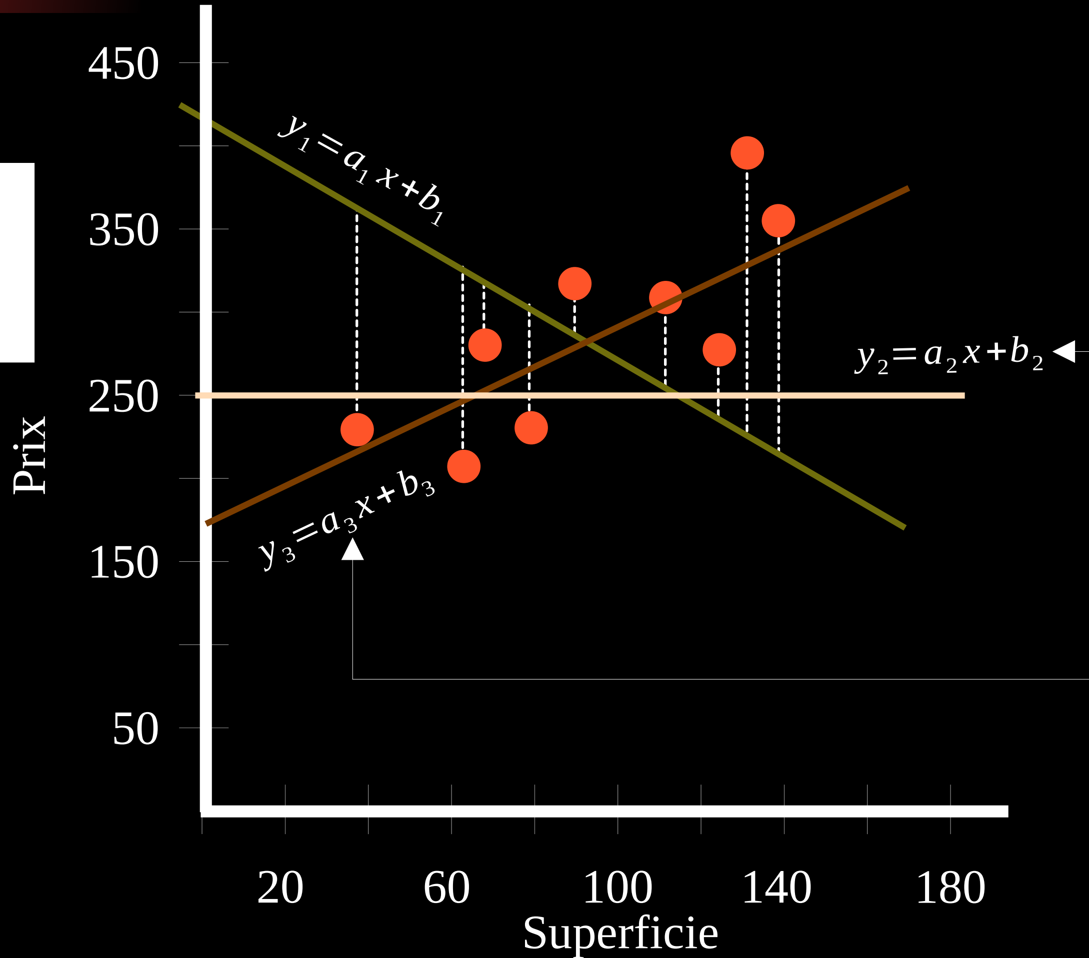
$$E = \; + \; + \; + \cdots$$

$$E=\left(e_1+e_2+\ldots+e_9\right)/I$$

I=9, est le nombre de points.

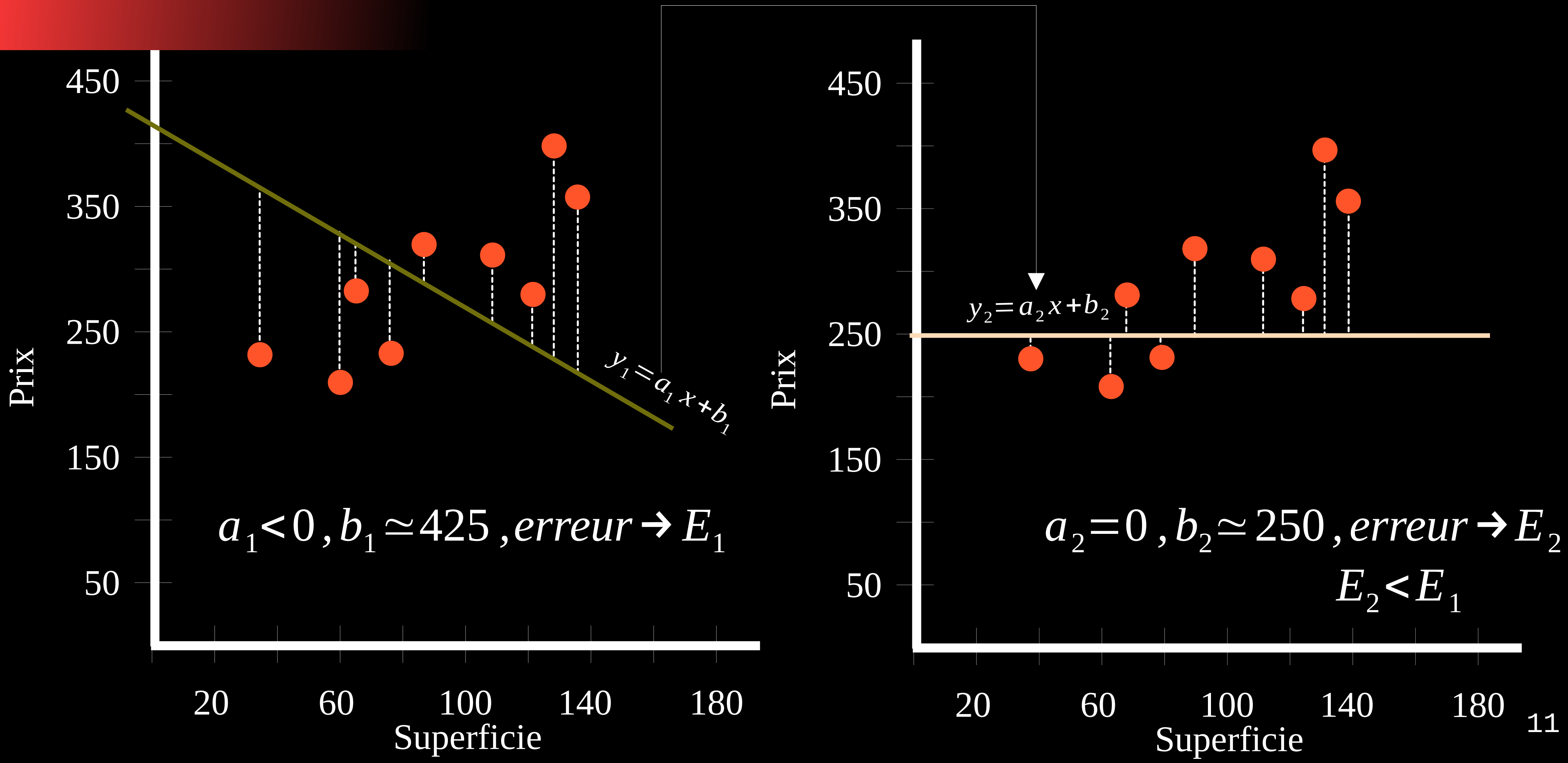
Theory

Graphique des prix (en XAF) de location des maisons en fonction de leur superficie (en m²).



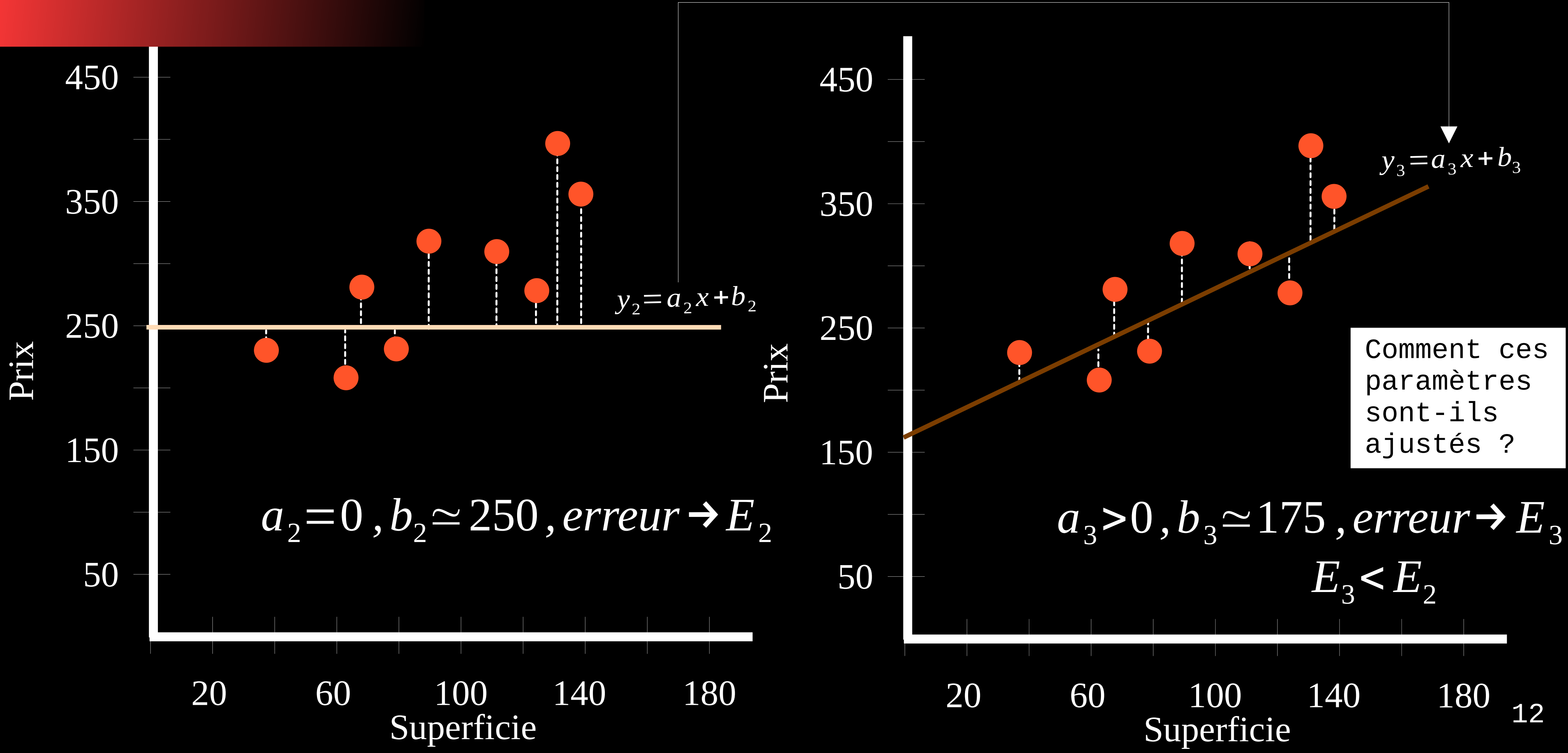
Modèles candidats (fonctions linéaires) pour mininimiser l’erreur.

Amélioration du modèle après n entraînements.



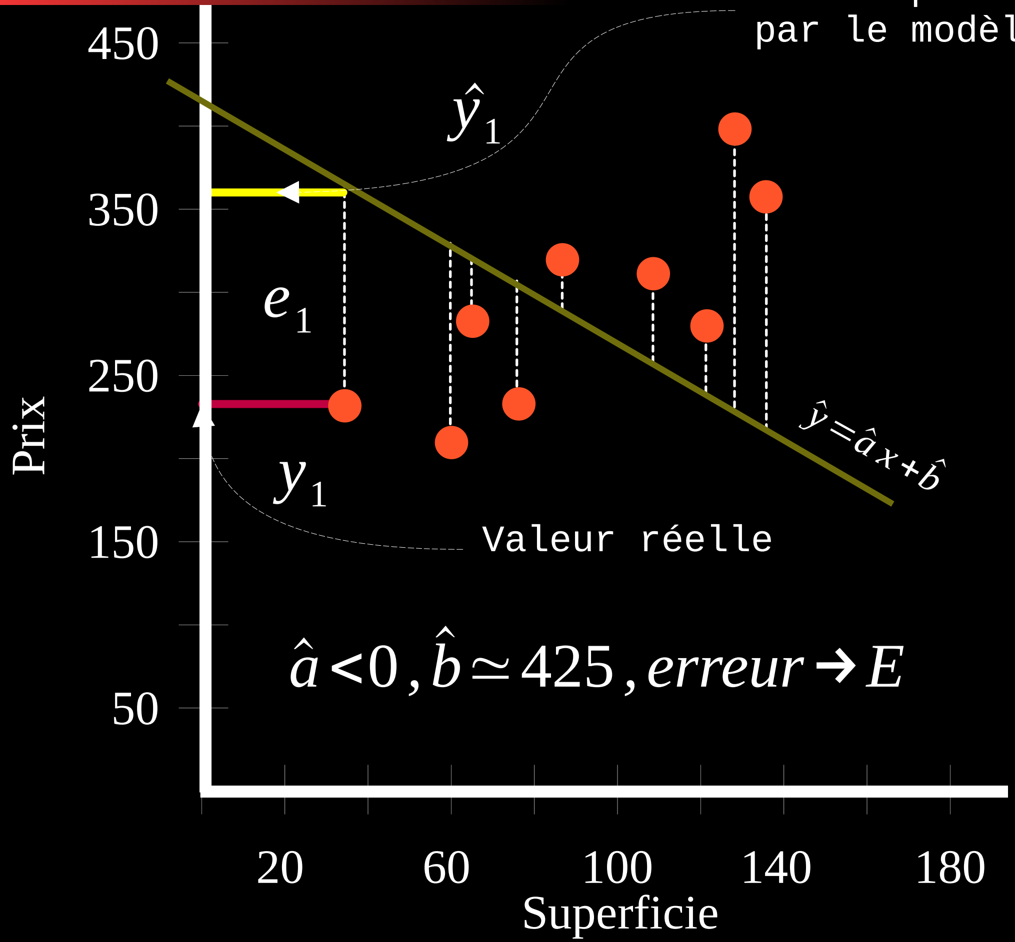
Theory

Amélioration du modèle après n+k entraînements.



Comment ces paramètres sont-ils ajustés ?

Theory



$$e_1=(y_1-\hat{y}_1)^2=[y_1-(\hat{a}x_1+\hat{b})]^2$$

$$E=(e_1+e_2+\ldots+e_9)/I=\frac{1}{I}\sum_{i=1}^{I} e_i$$

E est appellée **Mean Squared Error (MSE)** et I=9, est le nombre de points

$$E=\frac{1}{I}\sum_{i=1}^{I}[y_i-(\hat{a}x_i+\hat{b})]^2$$

Comment minimiser l'erreur E?

En trouvant les valeurs optimales de â et ̂b qui minimise cette erreur.

Dérivons E pour trouver les paramètres optimaux.

Theory

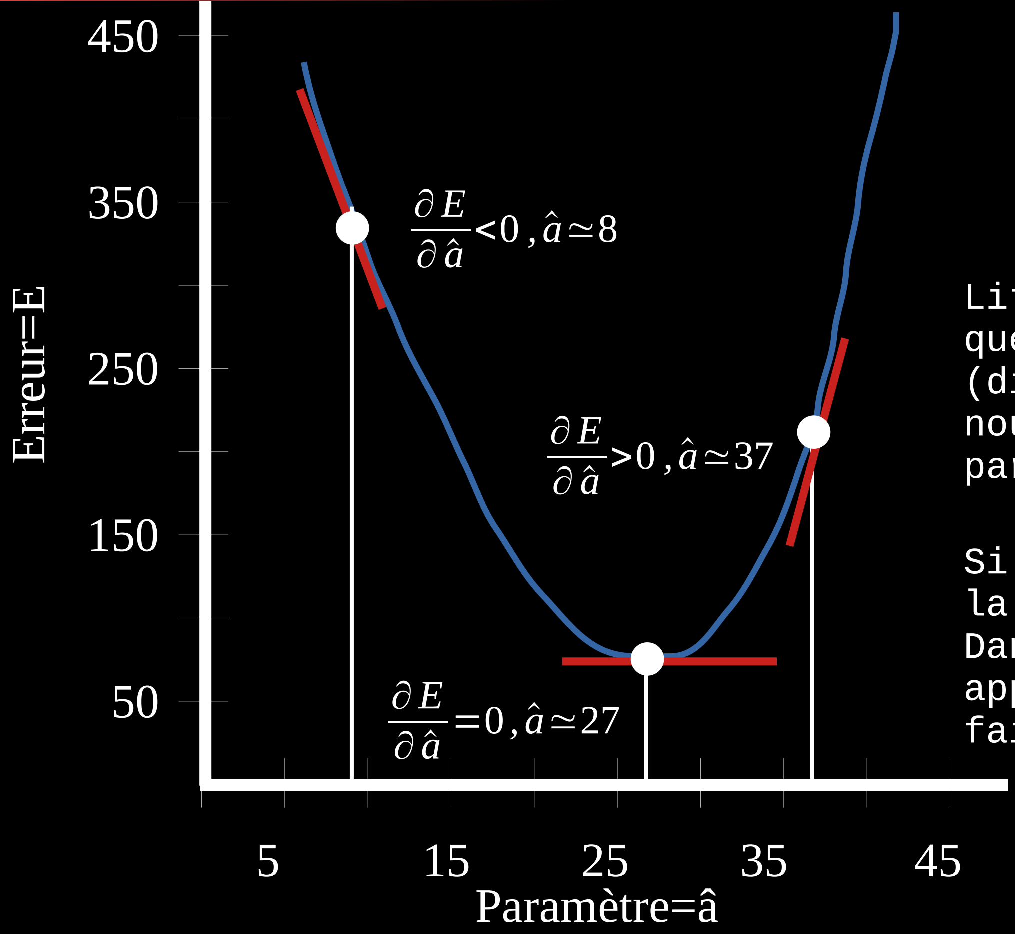
$$E = \frac{1}{I} \sum_{i=1}^{I} [y_i - (\hat{a} x_i + \hat{b})]^2$$

$$[(u-v)^2]' = 2(u-v)(u'-v'), u = y_i \text{ and } v = (\hat{a} x_i + \hat{b})$$

$$\frac{\partial E}{\partial \hat{a}} = \frac{1}{I} \sum_{i=1}^{I} 2[y_i - (\hat{a} x_i + \hat{b})](-x_i) = \frac{-2}{I} \sum_{i=1}^{I} [y_i - (\hat{a} x_i + \hat{b})](x_i)$$

$$\frac{\partial E}{\partial \hat{b}} = \frac{1}{I} \sum_{i=1}^{I} 2[y_i - (\hat{a} x_i + \hat{b})](-1) = \frac{-2}{I} \sum_{i=1}^{I} [y_i - (\hat{a} x_i + \hat{b})]$$

Rappel sur les dérivés des fonctions.



Supposons que nous voulons calculer la dérivé de E au point â=x, formellement on aura:

$$E'(\hat{a}=x)=\frac{\partial E}{\partial \hat{a}}$$

Littéralement, la fonction ci-dessus signifie: quelle est le taux de changement de E (diminution, augmentation, stagnation), lorsque nous sommes au point â. Ce taux est représenté par une droite tangente à la courbe de E.

Si nous voulons minimiser E, nous devons trouver la valeur â qui conduit à cette minimisation. Dans notre exemple, cette valeur est approximativement égale à 27. Ceci doit aussi se faire pour ƀ.

Calculons la dérivée de E en fonction de â et ƀ.

Rappel algèbre linéaire.

Addition de vecteurs.

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix} = \begin{bmatrix} a_1+b_1 \\ a_2+b_2 \\ \vdots \\ a_k+b_k \end{bmatrix} \text{ Exemple } \rightarrow \begin{bmatrix} 2.5 \\ 18.0 \\ \vdots \\ 7.5 \end{bmatrix} + \begin{bmatrix} 3.9 \\ 8.1 \\ \vdots \\ 4.2 \end{bmatrix} = \begin{bmatrix} 6.4 \\ 26.1 \\ \vdots \\ 11.7 \end{bmatrix}$$

Produit d'un nombre réel avec un vecteur.

$$X \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} Xa_1 \\ Xa_2 \\ \vdots \\ Xa_k \end{bmatrix} \text{ Exemple } \rightarrow 2 \begin{bmatrix} 3.9 \\ 8.1 \\ \vdots \\ 4.2 \end{bmatrix} = \begin{bmatrix} 7.8 \\ 16.2 \\ \vdots \\ 8.4 \end{bmatrix}$$

Rappel algèbre linéaire.

Pour multiplier deux matrices, le nombre de colonnes de la première matrice doit être égal au nombre de lignes de la seconde.

Produit de matrices.

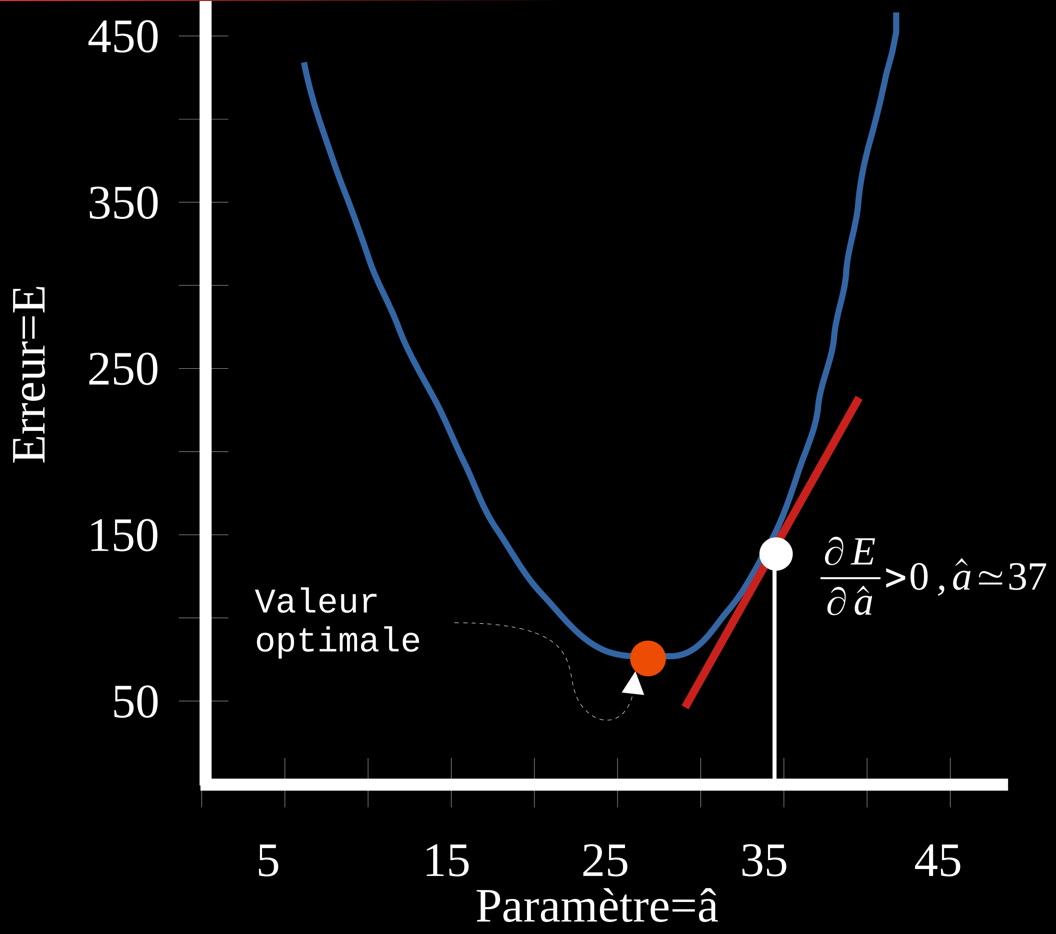
$$\underset{2 \times 2}{\begin{bmatrix} a_1^1 & a_2^1 \\ a_1^2 & a_2^2 \end{bmatrix}} \underset{2 \times 3}{\begin{bmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{bmatrix}} = \underset{2 \times 3}{\begin{bmatrix} a_1^1 b_1^1 + a_2^1 b_1^2 & a_1^1 b_2^1 + a_2^1 b_2^2 & a_1^1 b_3^1 + a_2^1 b_3^2 \\ a_1^1 b_1^1 + a_2^1 b_1^2 & a_1^1 b_2^1 + a_2^1 b_2^2 & a_1^1 b_3^1 + a_2^1 b_3^2 \end{bmatrix}}$$

Exemple

$$\begin{bmatrix} 2{,}1 & 4{,}3 \\ 1{,}7 & 7{,}0 \end{bmatrix} \begin{bmatrix} 2{,}5 & 10{,}0 & 13{,}2 \\ 5{,}4 & 4{,}4 & 6{,}5 \end{bmatrix} = \begin{bmatrix} 2{,}1*2{,}5+4{,}3*5{,}4 & 2{,}1*10{,}0+4{,}3*4{,}4 & 2{,}1*13{,}2+4{,}3*6{,}5 \\ 1{,}7*2{,}5+7{,}0*5{,}4 & 1{,}7*10{,}0+7{,}0*4{,}4 & 1{,}7*13{,}2+7{,}0*6{,}5 \end{bmatrix}$$

$$\begin{bmatrix} 2{,}1 & 4{,}3 \\ 1{,}7 & 7{,}0 \end{bmatrix} \begin{bmatrix} 2{,}5 & 10{,}0 & 13{,}2 \\ 5{,}4 & 4{,}4 & 6{,}5 \end{bmatrix} = \begin{bmatrix} 28{,}47 & 39{,}92 & 55{,}67 \\ 42{,}05 & 47{,}8 & 67{,}94 \end{bmatrix}$$

Theory



La nouvelle valeur de â sera:

$$\hat{a} = \hat{a} - \backepsilon \frac{\partial E}{\partial \hat{a}}$$

$\backepsilon > 0$ Est appelée **learning rate**.
Elle est souvent égale à 0.001

Ce processus d'ajustement appelé **descente de gradient** est répété n fois jusqu'à ce que l'on trouve la valeur optimale de â (ansi que ƀ) qui minimise E.

Que se passe-t'il si nous avons plusieurs valeurs observées ?

Theory

$$\hat{y}=\hat{a}_1 x_1+\hat{a}_2 x_2+\ldots+\hat{a}_k+x_k+\hat{b}=[x_1,x_2,\ldots,x_k]\begin{bmatrix}\hat{a}_1\\\hat{a}_2\\\vdots\\\hat{a}_k\end{bmatrix}+\hat{b}$$

$$\hat{y}=X\theta+\hat{b}\mid X=[x_1,x_2,\ldots,x_k],\theta=\begin{bmatrix}\hat{a}_1\\\hat{a}_2\\\vdots\\\hat{a}_k\end{bmatrix}$$

$$E=\frac{1}{I}\sum_{i=1}^{I}[y_i-(X_i\theta+\hat{b})]^2$$

$$X=[x_1,x_2,\ldots,x_k]\rightarrow X^T=\begin{bmatrix}x_1\\x_2\\\vdots\\x_k\end{bmatrix}$$

$$\frac{\partial E}{\partial\theta}=\frac{1}{I}\sum_{i=1}^{I}2[y_i-(X_i\theta+\hat{b})](-X_i)^T=\frac{-2}{I}\sum_{i=1}^{I}[y_i-(X_i\theta+\hat{b})](X_i)^T$$

Theory

$$X=[x_1,x_2,\dots,x_k]\rightarrow X^T=\begin{bmatrix}x_1\\x_2\\\vdots\\x_k\end{bmatrix}$$

$$\alpha_i=[y_i-(X_i\theta+\hat{b})]$$

$$\frac{\partial E}{\partial\theta}=\frac{-2}{I}\sum_{i=1}^{I}[y_i-(X_i\theta+\hat{b})](X_i)^T=\frac{-2}{I}\sum_{i=1}^{I}\alpha_i\begin{bmatrix}x_1^i\\x_2^i\\\vdots\\x_k^i\end{bmatrix}$$

$$\color{red}{\theta}=\theta-\ni\frac{\partial E}{\partial\theta}=\theta-\ni\frac{-2}{I}\sum_{i=1}^{I}\alpha_i\begin{bmatrix}x_1\\x_2\\\vdots\\x_k\end{bmatrix}=\begin{bmatrix}\hat{a}_1\\\hat{a}_2\\\vdots\\\hat{a}_k\end{bmatrix}-\ni\frac{-2}{I}\sum_{i=1}^{I}\alpha_i\begin{bmatrix}x_1^i\\x_2^i\\\vdots\\x_k^i\end{bmatrix}$$

Use Case

Cloud computing avec Google Colab

Configuration de l'environnement de travail avec du Bash Script.

Nous allons coder en Python.

Prédiction du prix de location en fonction de la superficie, le nombre de pièces, le quartier, etc.

Éditeurs de code : Visual Studio & Jupyter Notebook.

Versioning du modèle.

Use Case

Brève introduction à Python

- *Python est un langage de programmation codé en C;*
- *Langage impératif et interprété;*
- *Langage le plus utilisé en data science;*
- *Inclut diverses bibliothèques telles que Pandas pour la manipulation de données structurées, Numpy pour les calculs mathématiques, etc.*

https://www.datacamp.com/fr/blog/top-programming-languages-for-data-scientists-in-2022

Use Case

Use Case