

# Fouille de Données

# Data Mining

## **La Régression**

# Plan du cours

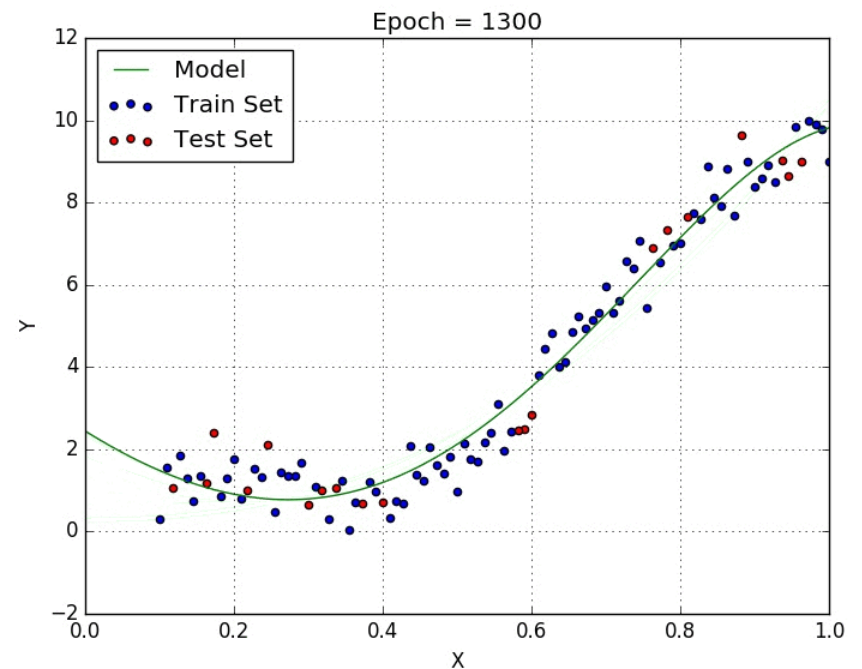
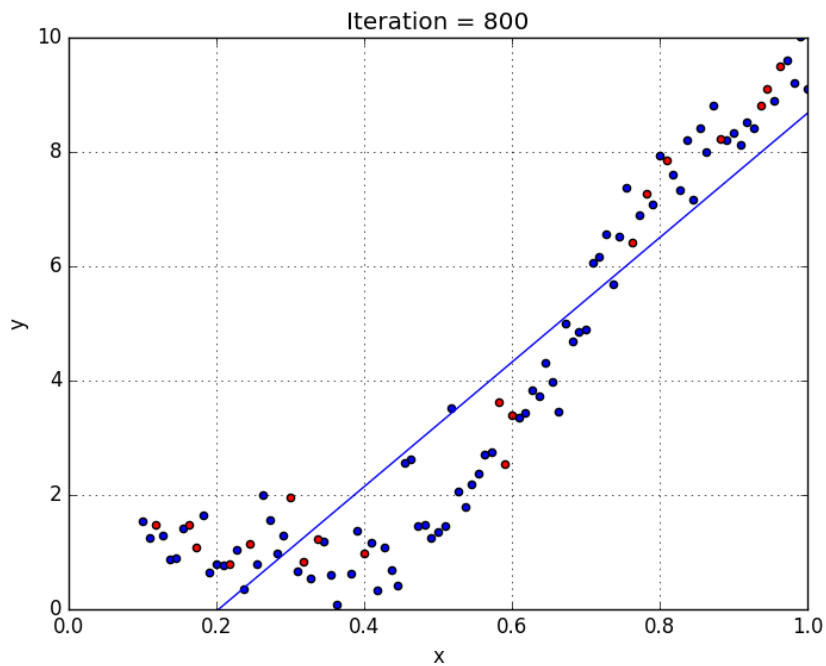
1. Régression / Estimation : Définition et principe
2. Régression linéaire simple.
3. Régression linéaire simple en utilisant le Gradient Descent.
4. Régression linéaire multiple
5. Régression polynomiale

# La régression

- La **régression** est la méthode utilisée pour l'estimation des valeurs **continues**.
- Son objectif est de trouver le meilleur modèle qui décrit la relation entre une variable continue de sortie et une ou plusieurs variables d'entrée.
- Prédire la valeur continue de la sortie **Y** selon une entrée **X** ou plusieurs entrées **X<sub>i</sub>** (attributs). = Expliquer une variable *Y* à l'aide d'une variable *X* ou plusieurs variables *X<sub>i</sub>*.
- Ex : prédire le cours de la bourse, le prix d'un appartement, ou bien l'évolution de la température sur Terre.
- Il s'agit donc de trouver une **fonction f** (=le modèle de régression) qui se rapproche le plus possible d'un scénario donné d'entrées et de sorties.
- Différents types de régression : linéaire, polynomiale, logistique (classification), Lasso, etc.

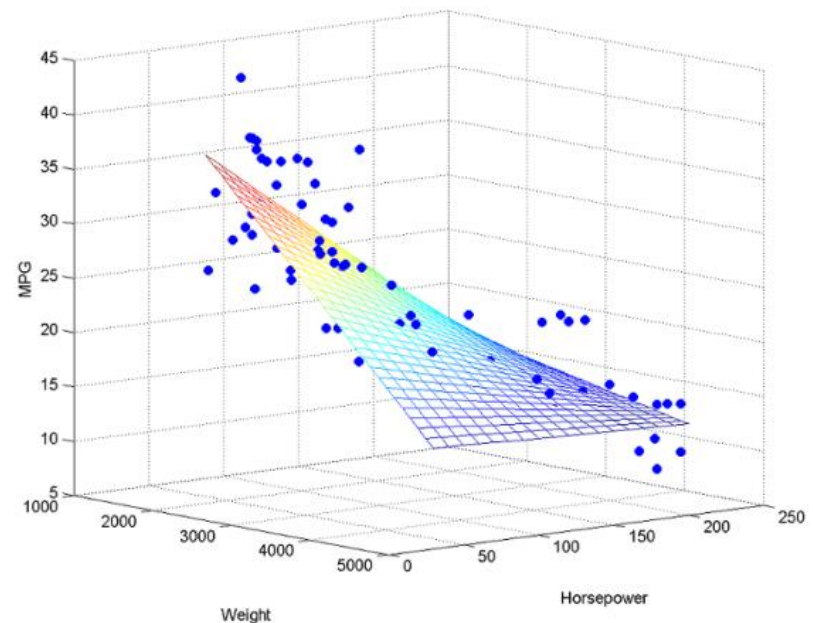
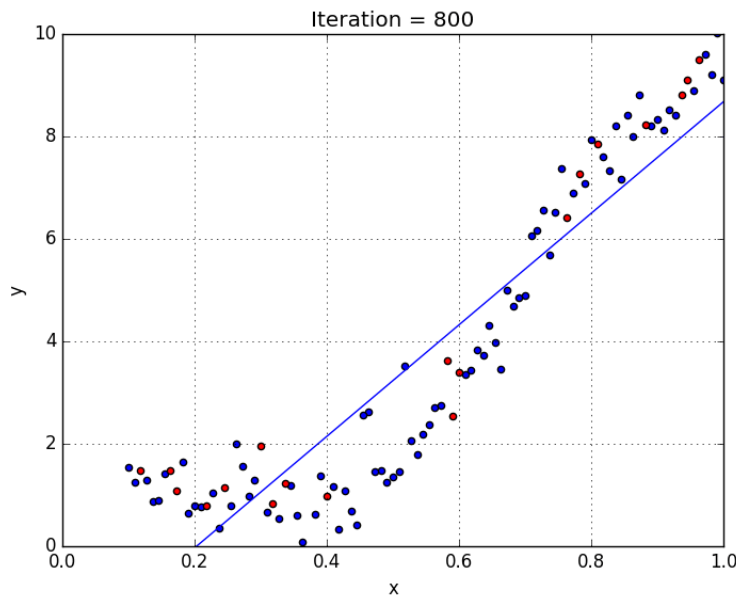
# La régression

- Différents types de régression : **linéaire**, **polynomiale**.



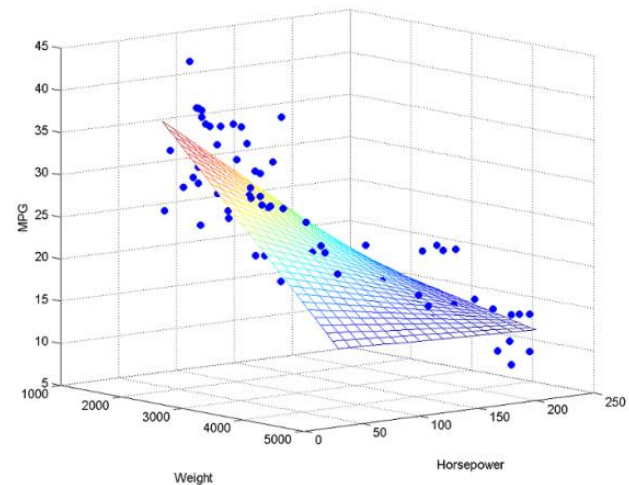
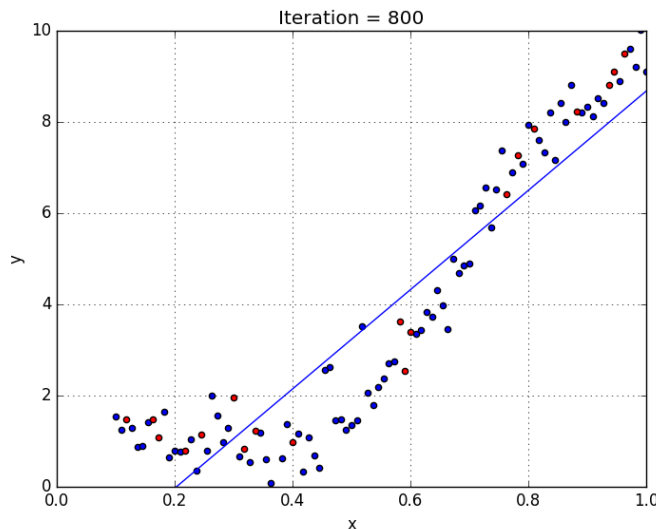
# La régression

- Régression **linéaire** : fonction  $f$  (modèle de régression) linéaire.
- Peut être : **Simple** ou **Multiple**.
- **Linéaire Simple** : utilisée pour estimer une sortie  $Y$  en fonction d'une seule entrée  $X$ .
- **Linéaire Multiple** : utilisée pour estimer une sortie  $Y$  en fonction de plusieurs entrées  $X_i$ .



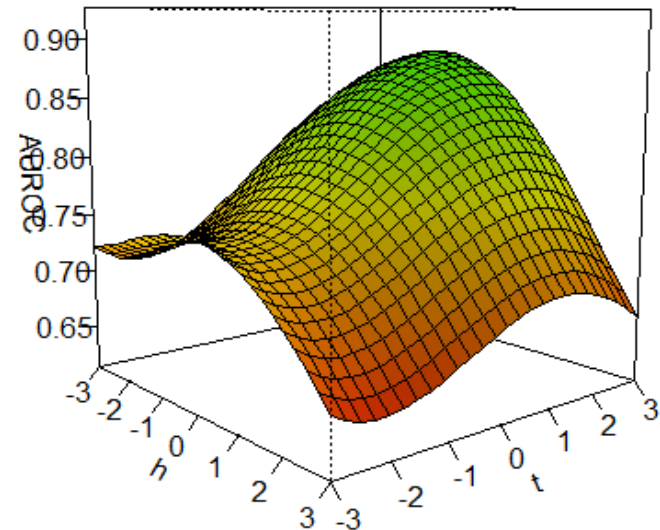
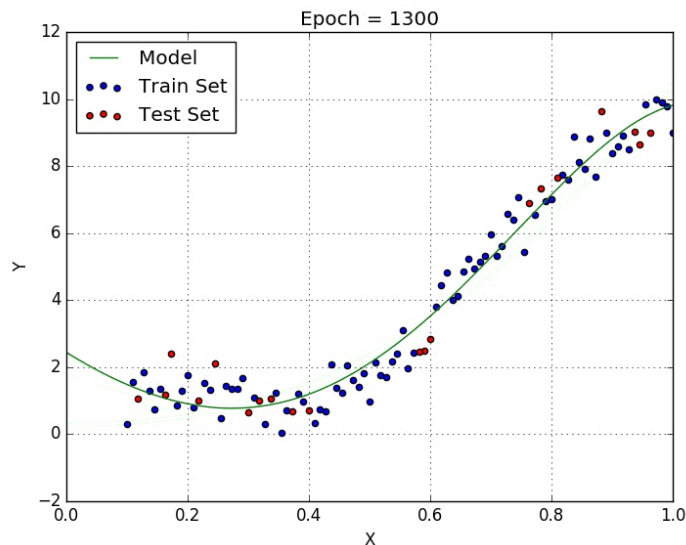
# La régression

- Régression **linéaire** : fonction  $f$  (modèle) linéaire.
- Peut être : **Simple** ou **Multiple**.
- **Linéaire Simple** : **Ex** : prédire le prix de vente d'un appartement (**Y**) en fonction de la surface habitable (**X**).
- **Linéaire Multiple** : **Ex** : prédire le prix de vente d'un appartement (**Y**) en fonction de la surface habitable (**X<sub>1</sub>**) et du nombre de pièces (**X<sub>2</sub>**) .



# La régression

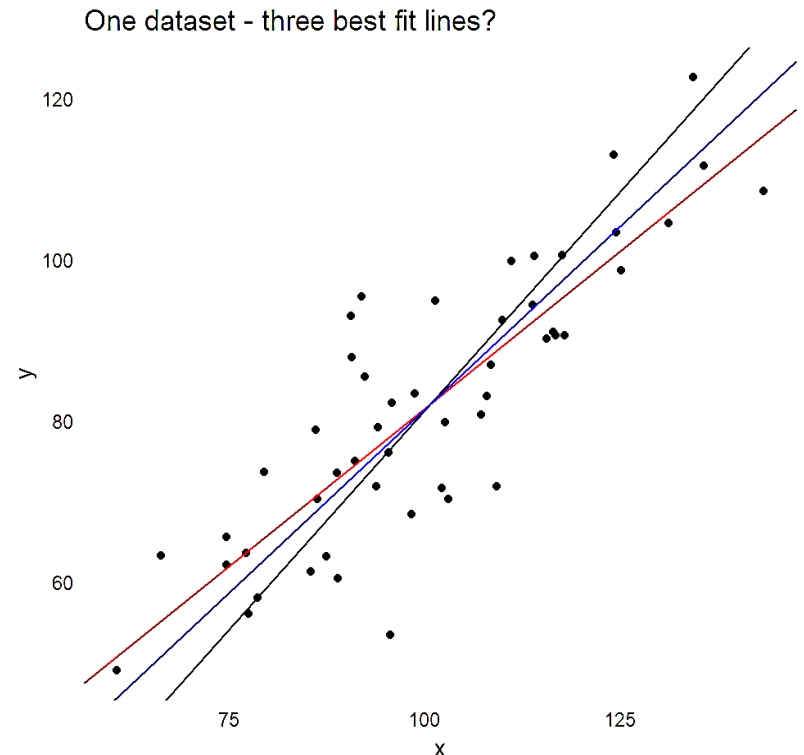
- Régression **polynomiale**: fonction  $f$  (modèle de régression) non linéaire.
- Peut être : **Simple** ou **Multiple**.
- **Polynomiale Simple** : utilisée pour estimer une sortie  $Y$  en fonction d'une seule entrée  $X$ .
- **Polynomiale Multiple** : utilisée pour estimer une sortie  $Y$  en fonction de plusieurs entrées  $X_i$ .



# La régression linéaire simple

## Régression linéaire simple

- **But** : Trouver un **modèle linéaire**  $f(x)=b_1x+b_0$  où  $b_1$  et  $b_0$  sont les **paramètres/coefficients** du modèle.
- $y = b_1x + b_0$
- Trouver le **meilleur** modèle (Best fit line) =>
  - => Trouver les **meilleurs** (optimales) valeurs des paramètres  $b_1$  et  $b_0$ .
  - => Faire le **minimum** d'erreurs possible sur les prédictions de Y.





# La régression linéaire simple

## Régression linéaire simple – Etapes de base

### ▪ **Exemple :**

- **X** = nombre d'heures passées à réviser
- **Y** = Note étudiant (/100)

### **Objectif :**

On souhaite savoir si, de façon générale, le nombre d'heures passées à réviser a une **influence** sur la note obtenue et sous quelle forme cette influence peut être exprimée.

Le but est d'**expliquer** au mieux comment la note d'un étudiant varie en fonction du nombre d'heures de révision et éventuellement de **prédire** la note à partir d'un nombre d'heures donné.

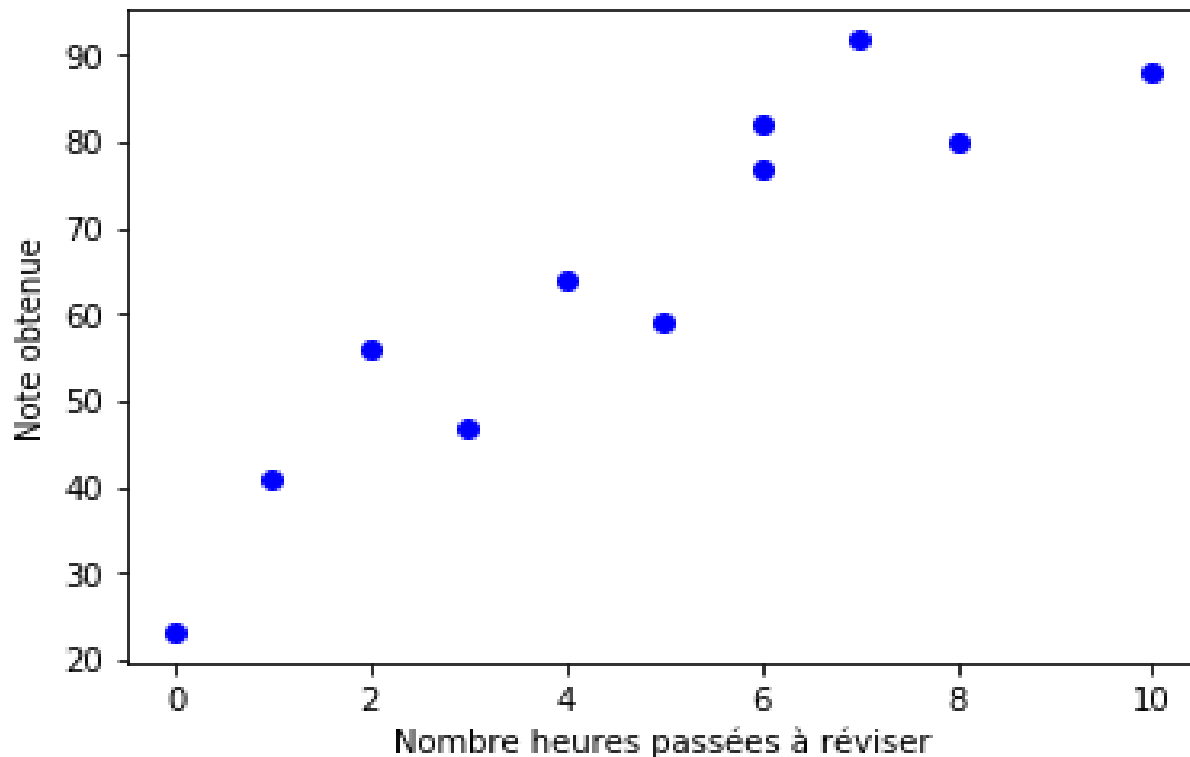
Hours spent on essay	Grade
-------------------------------	-------

6	82
10	88
2	56
4	64
6	77
7	92
0	23
1	41
8	80
5	59
3	47

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

- **Exemple :**  $X$  = nombre d'heures de révision ----  $Y$  = Note étudiant



Hours spent on essay	Grade
6	82
10	88
2	56
4	64
6	77
7	92
0	23
1	41
8	80
5	59
3	47

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

### ▪ Trouver la droite : $y = b_1x + b_0$

**1** - Calculer les valeurs de  **$b_1$**  (le slope) et  **$b_0$**  (l'intercept) selon :

$$\left\{ \begin{array}{l} b_1 = \frac{\sum (x - \bar{x}) * (y - \bar{y})}{\sum (x - \bar{x})^2} = \frac{\text{Covariance}(x,y)}{\text{Variance}(x)} \\ b_0 = \bar{y} - b_1 \bar{x} \end{array} \right.$$

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

### ▪ Trouver la droite : $y = b_1x + b_0$

1 - Calculer les valeurs de **b<sub>1</sub>** (le slope) et **b<sub>0</sub>** (l'intercept) :

$$b_1 = \frac{\sum (x - \bar{x}) * (y - \bar{y})}{\sum (x - \bar{x})^2} = \frac{Covariance(x,y)}{Variance(x)}$$

$$b_0 = \bar{y} - b_1 \bar{x}$$

	Hours spent on essay	Grade
	6	82
	10	88
	2	56
	4	64
	6	77
	7	92
	0	23
	1	41
	8	80
	5	59
	3	47
Mean	4.72	64.45
	$\bar{x}$	$\bar{y}$

# La régression linéaire simple

Mean 4.72 64.45  
 $\bar{x}$   $\bar{y}$

1 - Calculer les valeurs de **b1** (le slope) et **b0** (l'intercept) :

Hours spent on essay	Grade	Hours spent – Average Hours Spent $(x - \bar{x})$	Grade – Average Grade $(y - \bar{y})$	$(x - \bar{x}) \times (y - \bar{y})$
6	82	1.27	17.55	22.33
10	88	5.27	23.55	124.15
2	56	-2.73	-8.45	23.06
4	64	-0.73	-0.45	0.33
6	77	1.27	12.55	15.97
7	92	2.27	27.55	62.60
0	23	-4.73	-41.45	195.97
1	41	-3.73	-23.45	87.42
8	80	3.27	15.55	50.88
5	59	0.27	-5.45	-1.49
3	47	-1.73	-17.45	30.15

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

### ▪ Trouver la droite : $y = b_1x + b_0$

1 - Calculer les valeurs de **b1** (le slope) et **b0** (l'intercept) :

$$\Sigma(x - \bar{x}) * (y - \bar{y}) = 611.36$$

$$\Sigma(x - \bar{x})^2 = 94.18$$

Mean	4.72	64.45
	$\bar{x}$	$\bar{y}$



$$b_1 = 611.36 / 94.18 \\ = 6.49$$

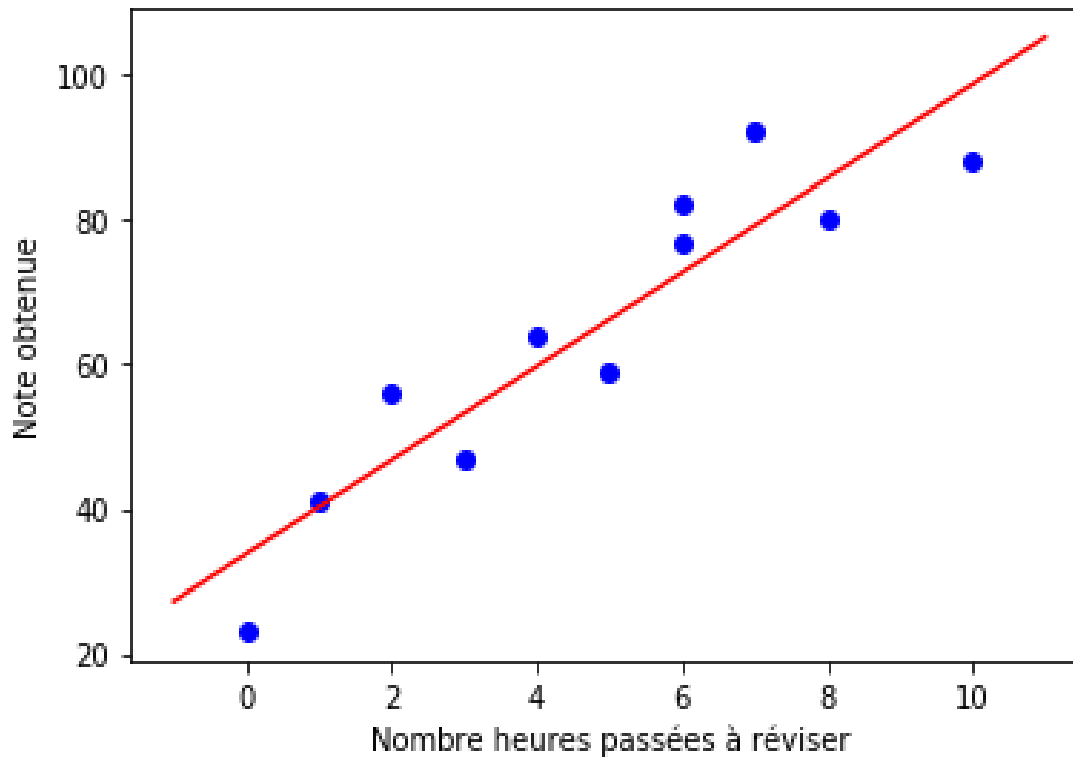
$$b_0 = 64.45 - (6.49 * 4.72) \\ = 33.81$$

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

- Trouver la droite :  $y = 6.49 * x + 33.81$

1 - Calculer les valeurs de **b1** (le slope) et **b0** (l'intercept) :



# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**2** – Prédire les nouvelles valeurs Y (la note) selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

Hours spent on essay	Grade	Predicted Grade
		72.716216
		98.681467
6	82	46.750965
10	88	59.733591
2	56	72.716216
4	64	79.207529
6	77	33.768340
7	92	40.259653
0	23	85.698842
1	41	66.224903
8	80	53.242278
5	59	
3	47	

Ex :

$$Y_1 = 6.49 * (6) + 33.81 = 72.716216$$



# La régression linéaire simple

## Régression linéaire simple – Etapes de base

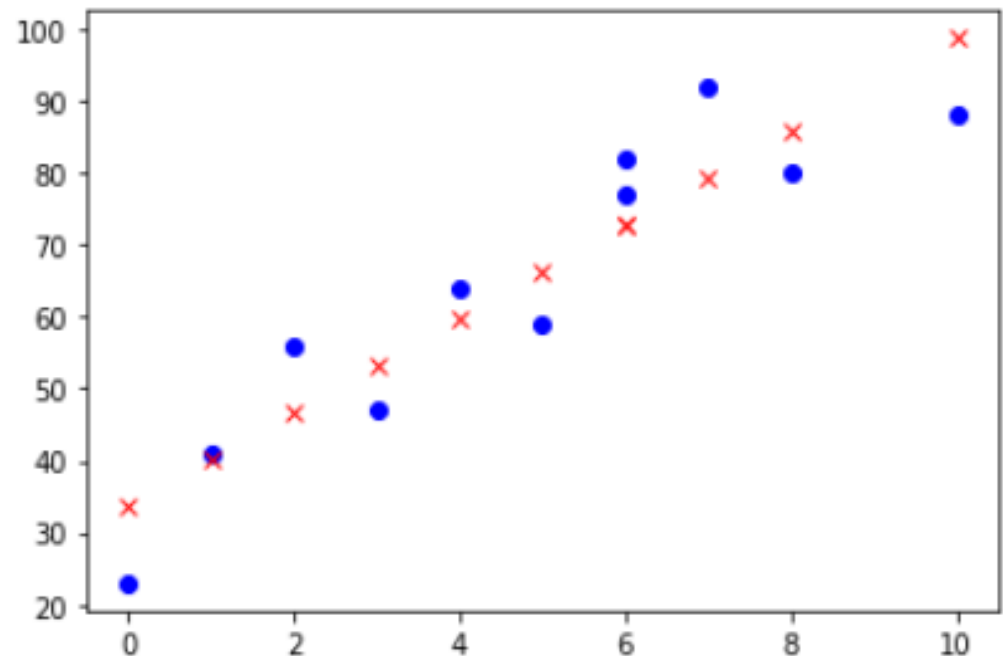
**2** – Prédire les nouvelles valeurs Y (la note) selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

En bleu : vraie valeur - **y**

En rouge : valeur prédite – **y\_pred**

Hours spent on essay	Grade	Predicted Grade
		72.716216
		98.681467
6	82	46.750965
10	88	59.733591
2	56	72.716216
4	64	79.207529
6	77	33.768340
7	92	40.259653
0	23	85.698842
1	41	66.224903
8	80	53.242278
5	59	
3	47	



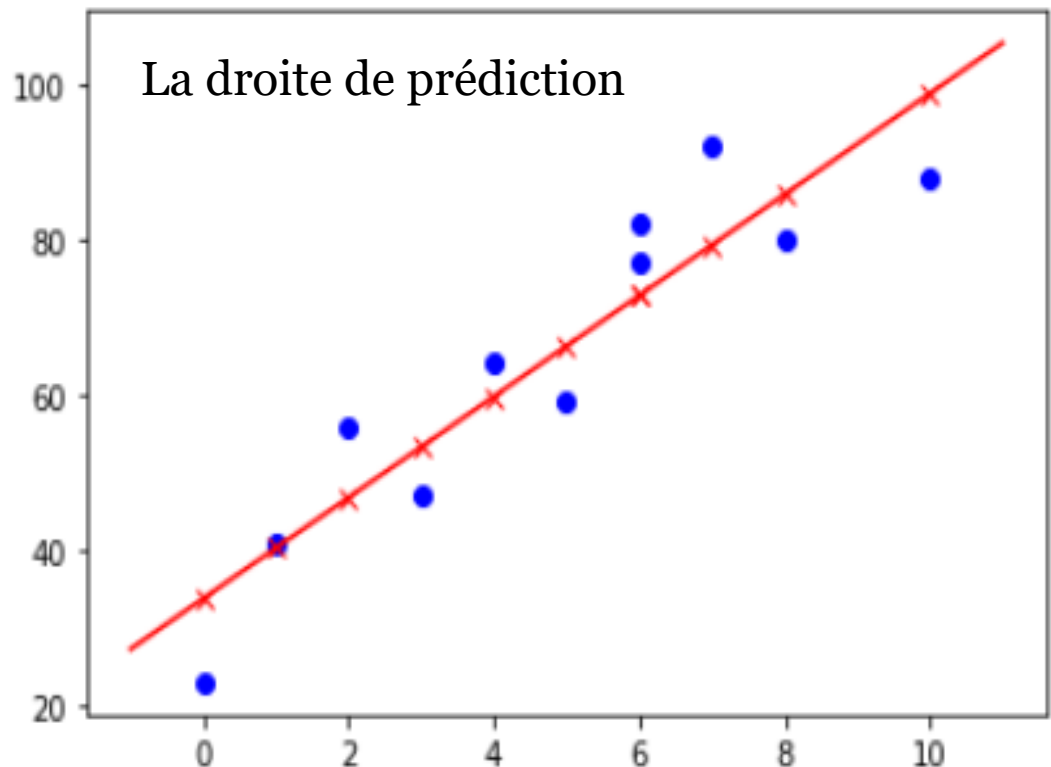
# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**2** – Prédire les nouvelles valeurs Y (la note) selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

Hours spent on essay	Grade	Predicted Grade
		72.716216
		98.681467
6	82	46.750965
10	88	59.733591
2	56	72.716216
4	64	79.207529
6	77	33.768340
7	92	40.259653
0	23	85.698842
1	41	66.224903
8	80	53.242278
5	59	
3	47	



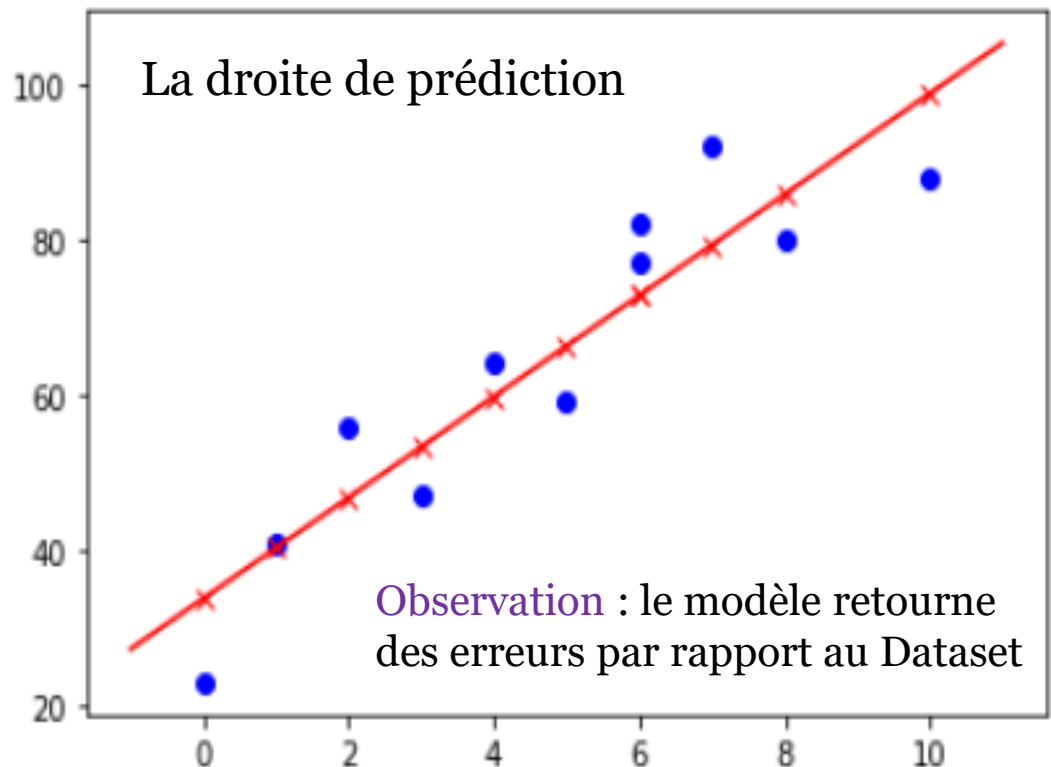
# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**2** – Prédire les nouvelles valeurs Y (la note) selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

Hours spent on essay	Grade	Predicted Grade
		72.716216
		98.681467
6	82	46.750965
10	88	59.733591
2	56	72.716216
4	64	79.207529
6	77	33.768340
7	92	40.259653
0	23	85.698842
1	41	66.224903
8	80	53.242278
5	59	
3	47	



# La régression linéaire simple

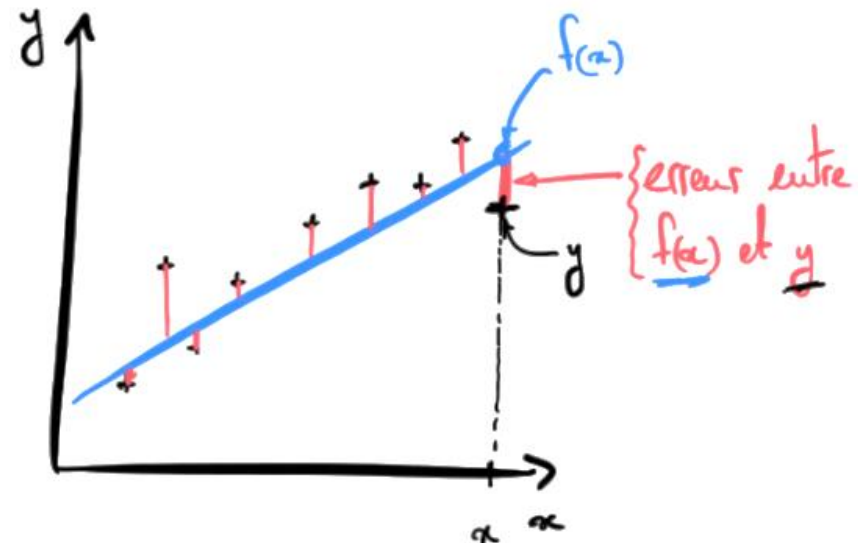
## Régression linéaire simple – Etapes de base

**3** – Evaluation du modèle de régression en estimant les **erreurs** sur les prédictions  $y_{\text{pred}}$  (i.e.  $f(x)$ ) :

Différentes **fonction de coût** permettant d'estimer l'erreur d'un modèle. Ex: **RMSE** – Root Mean Square Error / sqrt(l'erreur quadratique moyenne)

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_{\text{pred},i} - y_i)^2}{n}}$$

$$y = b_1x + b_0 + \text{err}$$

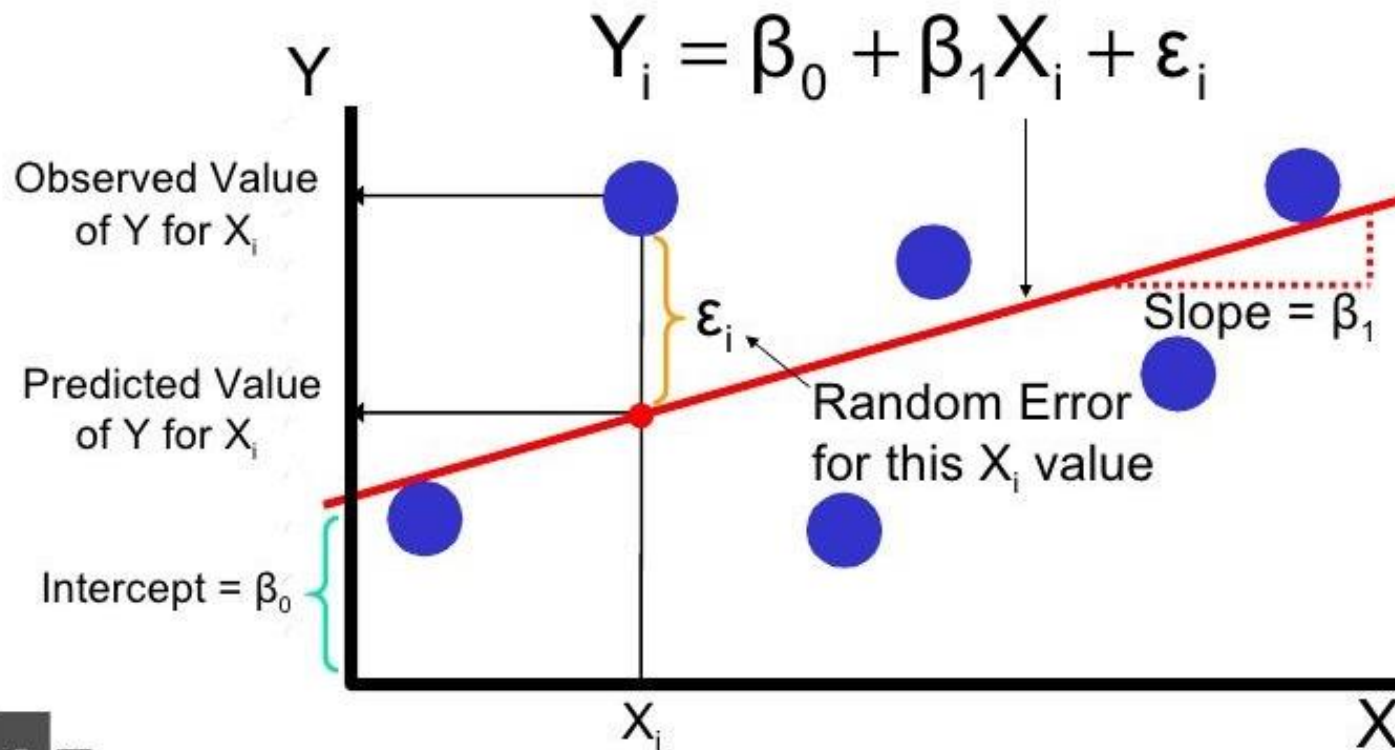


Fonction Coût = l'ensemble des erreurs.

# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**3** – Evaluation du modèle de régression en estimant les **erreurs** sur les prédictions  $y_{\text{pred}}$  (i.e.  $f(x)$ ) :



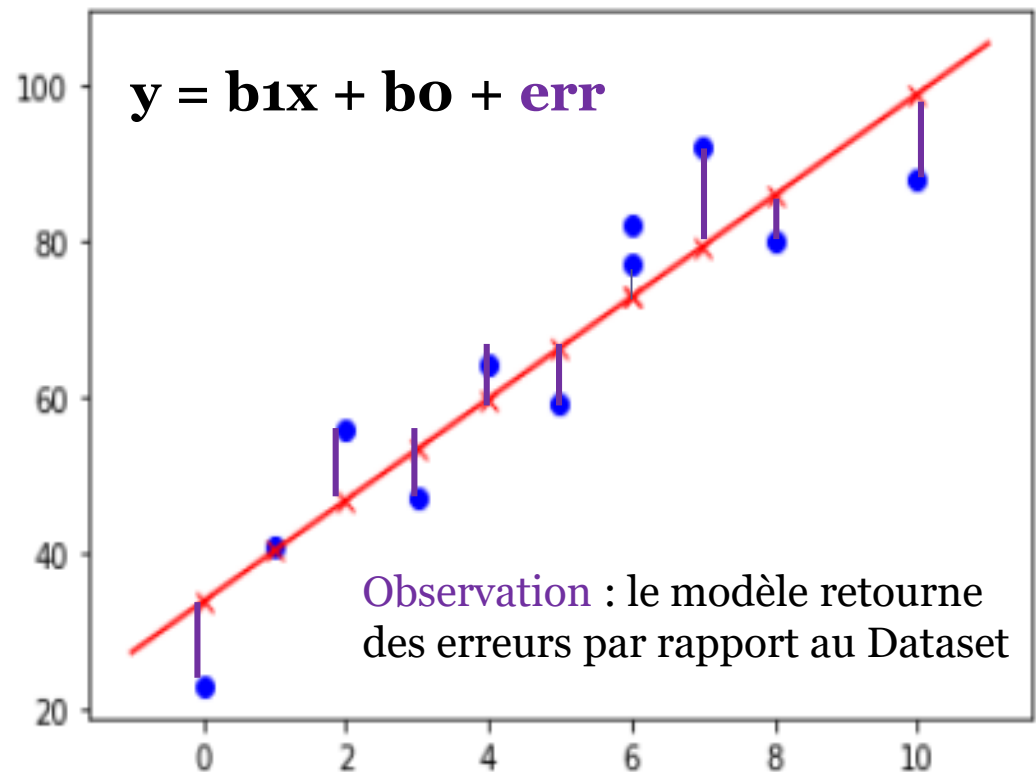
# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**3** – Evaluation du modèle de régression en estimant le total des **erreurs** sur les prédictions :

**Ex** : le modèle trouvé a fait une erreur **err** de **+5.698842** sur l'exemple (8, 80) du dataset :

$$(y_{\text{pred}} - y) = 85.698842 - 80 = +5.698842$$



# La régression linéaire simple

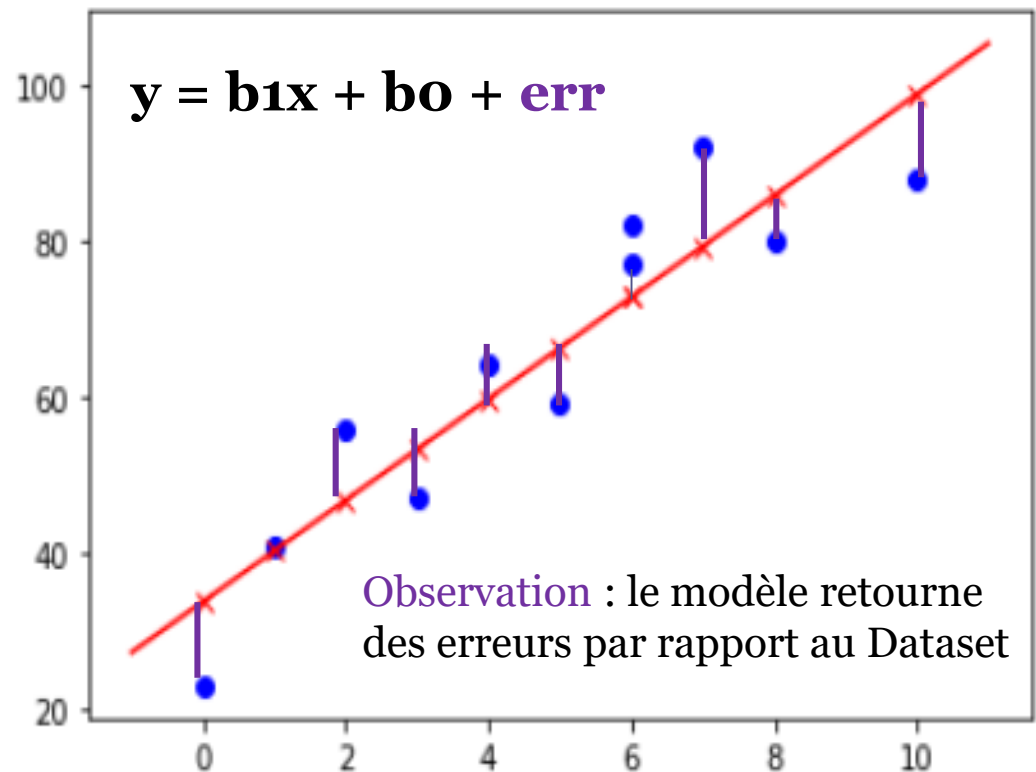
## Régression linéaire simple – Etapes de base

**3** – Evaluation du modèle de régression en estimant le total des **erreurs** sur les prédictions :

Chaque prédiction (exemple/datapoint) s'accompagne d'une erreur, on a donc ***n*** erreurs.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_{pred,i} - y_i)^2}{n}}$$

=> **RMSE** = 8.125



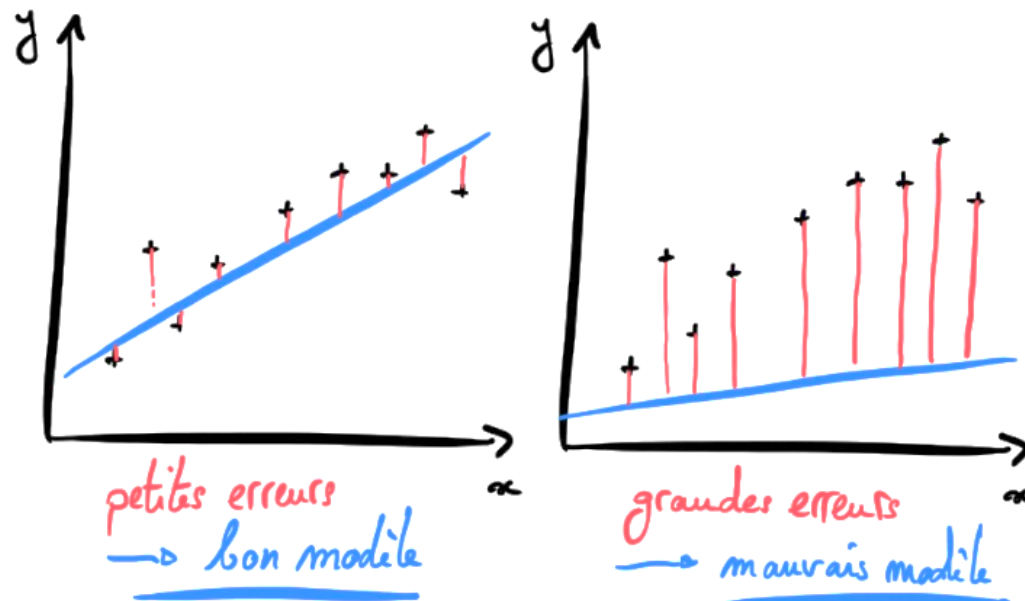
# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**4 – Minimiser l'erreur** en minimisant la fonction coût:

$y = b_1x + b_0 + \text{err}$  => Ramener **err** vers 0 => RMSE le plus petit possible

Le but est de trouver les meilleures (optimales) estimations des coefficients  $b_0$  et  $b_1$  pour minimiser les erreurs de prédiction de  $y$  à partir de  $x$ .





# La régression linéaire simple

## Régression linéaire simple – Etapes de base

**4 – Minimiser l'erreur** en minimisant la fonction coût:

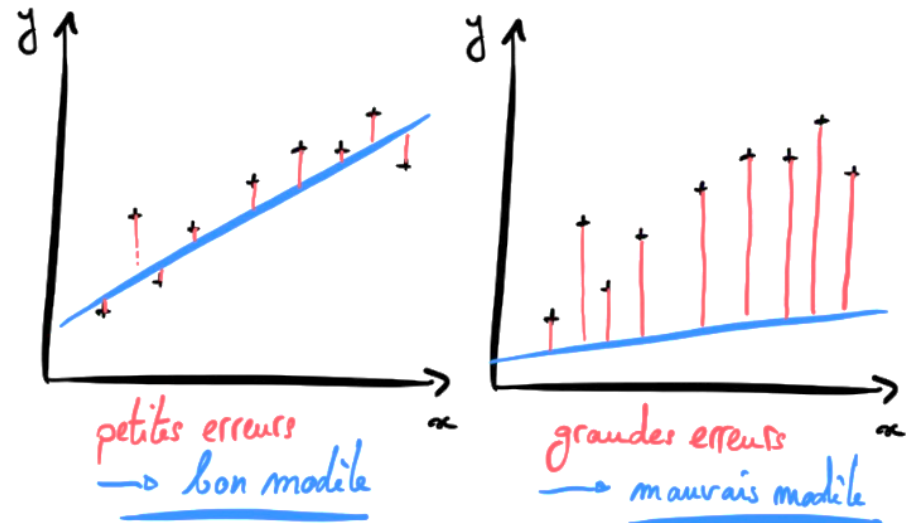
$$y = b_1x + b_0 + \text{err} \Rightarrow \text{Ramener err vers 0} \Rightarrow \text{RMSE le plus petit possible}$$

Le but est de trouver les meilleures (optimales) estimations des coefficients  $b_0$  et  $b_1$  pour minimiser les erreurs de prédiction de  $y$  à partir de  $x$ .

Minimiser l'erreur →

**Problème d'optimisation** →

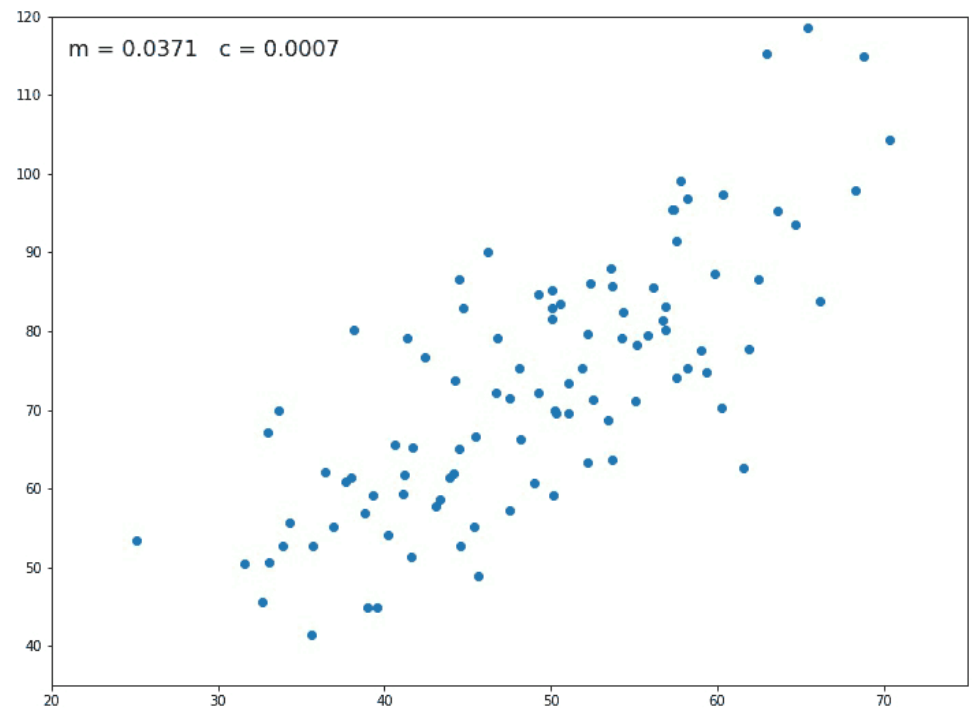
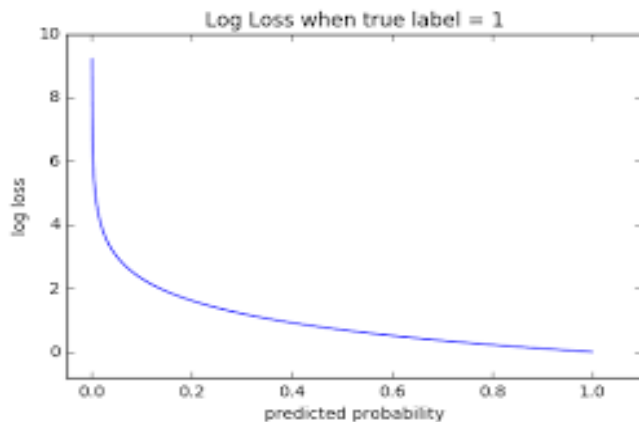
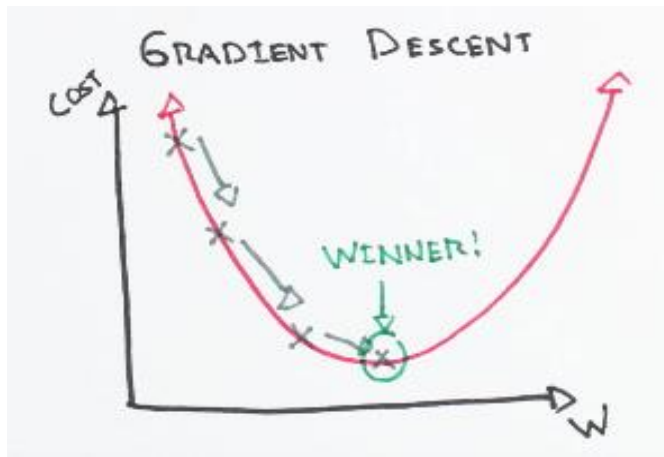
Itérer les étapes précédentes en utilisant un **algorithme d'optimisation** cherchant à minimiser la **fonction coût**.



# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

**Ex : Algorithme de Gradient Descent** (Descente du Gradient)

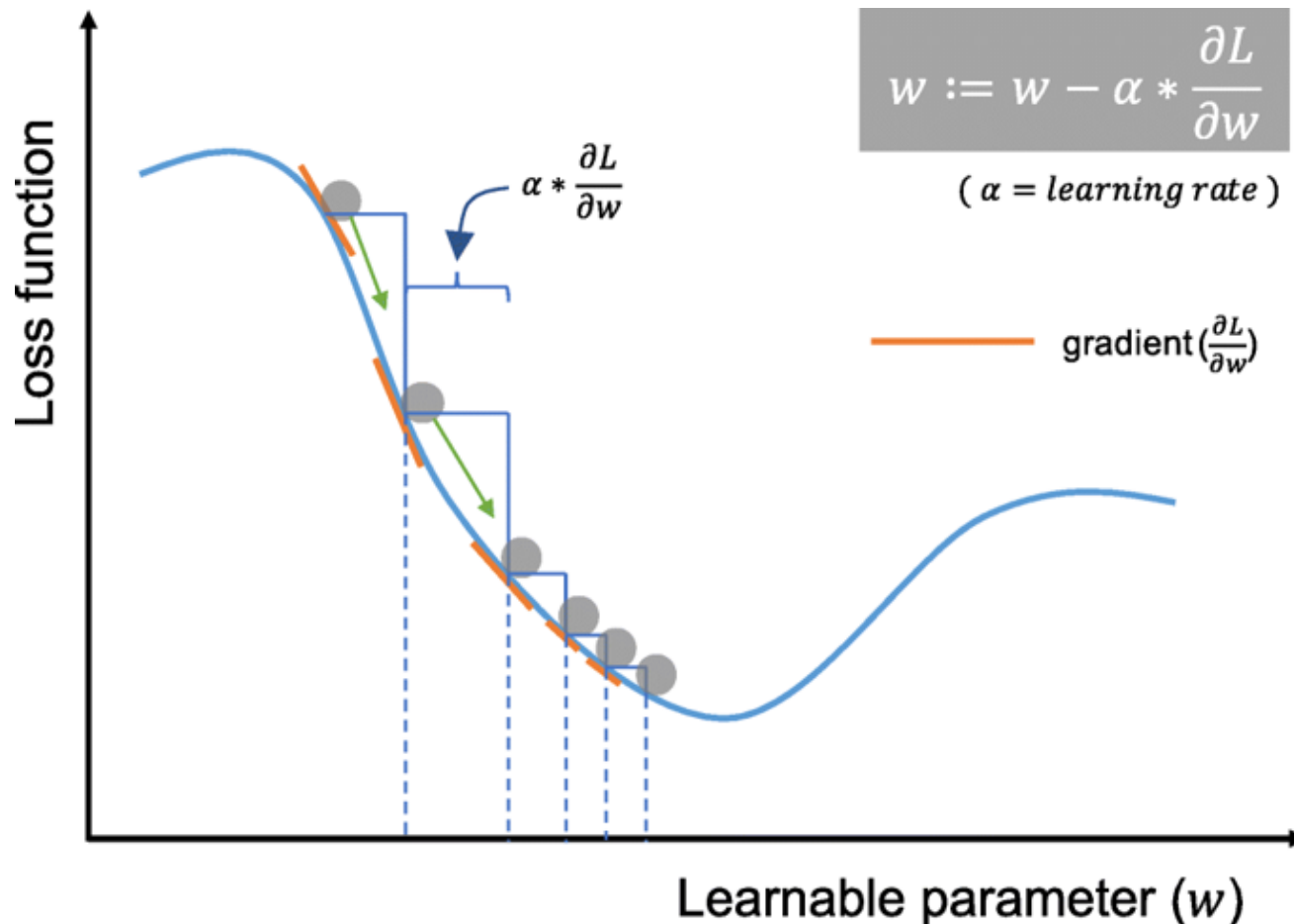


Gif : [https://miro.medium.com/max/1400/1\\*CjTBNFUEI\\_IokEOXJoozKw.gif](https://miro.medium.com/max/1400/1*CjTBNFUEI_IokEOXJoozKw.gif)

# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

Ex : **Algorithme de Gradient Descent** (Descente du Gradient)



# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

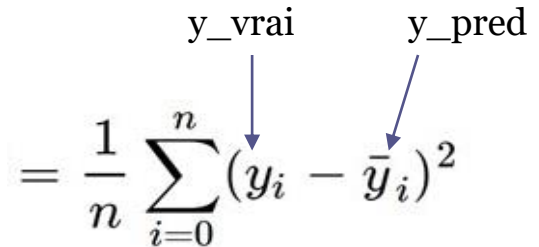
Ex : **Algorithme de Gradient Descent** (Descente du Gradient)

### Etapes :

- Modèle linéaire avec  $f(x)$  :  $y = m * x + c$

- Fonction cout : MSE – Mean Squared Error

$$= \frac{1}{n} \sum_{i=0}^n (y_i - \bar{y}_i)^2$$



1 - Initialiser les paramètres  $m$  et  $c$  à 0 :  $m = 0$  et  $c = 0$  ( $m$  est  $b_1$  et  $c$  est  $b_0$ )

2 - Choisir et fixer le nombre d'itération (**epochs**) et **learning\_rate**.

3 – Calculer les prédictions **y\_pred** pour chaque exemple dans le dataset, selon :  $y_{pred} = m * x + c$

4 – Calculer le gradient : i.e. les dérivées partielles **Dm** et **Dc**.

5 - Mettre à jour les valeurs de  $m$  et de  $c$  en fonction du gradient.

Répéter (3 4 5) *epochs* fois afin de le minimiser gradient et optimiser  $m$ ,  $c$

# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

**Ex : Algorithme de Gradient Descent** (Descente du Gradient)

### Etapes :

**4** – Calculer le gradient : i.e. les dérivées partielles  **$D_m$**  et  **$D_c$**  :

- **$D_m$**  : Dérivée partielle de la fonction coût selon le paramètre  $m$
- **$D_c$**  : Dérivée partielle de la fonction coût selon le paramètre  $c$

$$D_m = \frac{1}{n} \sum_{i=0}^n 2(y_i - (mx_i + c))(-x_i)$$

$$D_m = \frac{-2}{n} \sum_{i=0}^n x_i(y_i - \bar{y}_i)$$

$$D_c = \frac{-2}{n} \sum_{i=0}^n (y_i - \bar{y}_i)$$

# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

**Ex : Algorithme de Gradient Descent** (Descente du Gradient)

### Étapes :

**5** – Mettre à jour les valeurs de ***m*** et de ***c*** en fonction du gradient et du learning rate *L*, comme suit :

$$m = m - L \times D_m$$

$$c = c - L \times D_c$$

Répétez les étapes (3, 4, et 5) *epochs* fois; jusqu'à ce que la fonction de coût a une très petite valeur ou idéalement = 0 (ce qui signifie 0 erreur ou 100% de précision). Les dernières valeurs de ***m*** et de ***c*** trouvées représentent leurs valeurs optimales.

# La régression linéaire simple

## Régression linéaire simple – Utilisation d'un algorithme d'optimisation

**Ex : Algorithme de Gradient Descent** (Descente du Gradient)

Exécutions pour l'exemple précédent:

On pose : epochs = 11 et l\_rate = 0.01

```
>epoch=0, lrate=0.010, m=7.205455, c=1.289091
>epoch=1, lrate=0.010, m=9.834750, c=1.871157
>epoch=2, lrate=0.010, m=10.783632, c=2.192994
>epoch=3, lrate=0.010, m=11.115504, c=2.418682
>epoch=4, lrate=0.010, m=11.220881, c=2.608478
>epoch=5, lrate=0.010, m=11.243171, c=2.784517
>epoch=6, lrate=0.010, m=11.235038, c=2.954926
>epoch=7, lrate=0.010, m=11.215822, c=3.122697
>epoch=8, lrate=0.010, m=11.192622, c=3.288929
>epoch=9, lrate=0.010, m=11.168048, c=3.454030
>epoch=10, lrate=0.010, m=11.143055, c=3.618152
>epoch=11, lrate=0.010, m=11.117996, c=3.781355
```

# La régression linéaire multiple

Régression linéaire multiple :

- La régression linéaire multiple utilisée pour estimer une sortie en fonction de plusieurs entrées n'est qu'une extension de la précédente, on a donc :

$$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_n X_{ni} .$$

- Calcul et optimisation des paramètres du modèle :  $b_0, b_1, b_2, \dots, b_n$ .
- $X$  dans ce cas n'est plus un tableau mais une matrice de  $n$  attributs et  $m$  exemples d'entraînement.

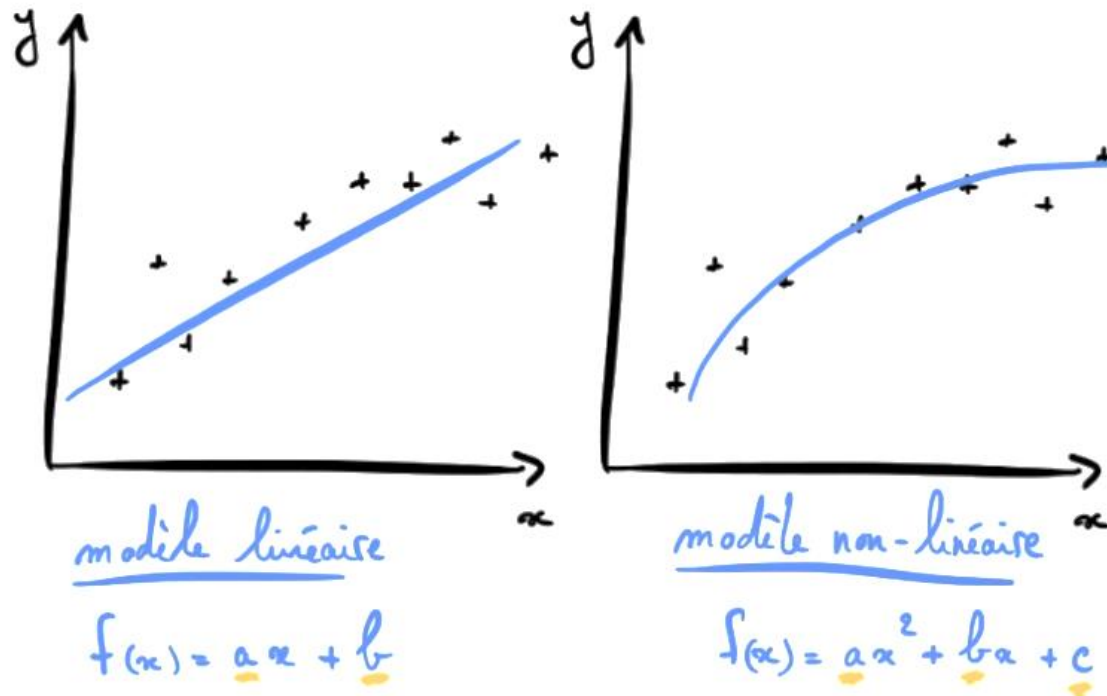


# La régression polynomiale

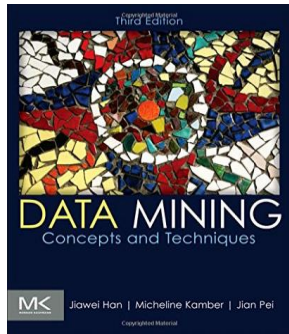
Régression polynomiale simple : Modèle non linéaire : Courbe

= Modèle **polynômial** de **degré 2** :  $f(x) = ax^2 + bx + c$

A trouver les paramètres optimaux  $a$ ,  $b$ , et  $c$ , en utilisant le Gradient Descent comme précédemment.

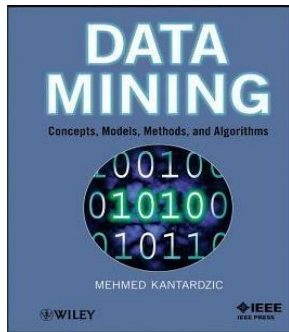


# Ressources



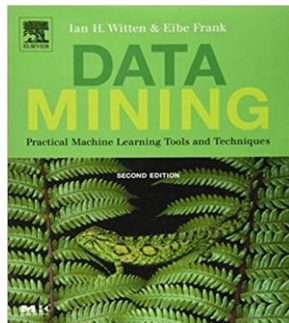
## **Data Mining : concepts and techniques, 3rd Edition**

- ✓ Auteur : Jiawei Han, Micheline Kamber, Jian Pei
- ✓ Éditeur : Morgan Kaufmann Publishers
- ✓ Edition : Juin 2011 - 744 pages - ISBN 9780123814807



## **Data Mining : concepts, models, methods, and algorithms**

- ✓ Auteur : Mehmed Kantardzi
- ✓ Éditeur : John Wiley & Sons
- ✓ Edition : Aout 2011 – 552 pages - ISBN : 9781118029121



## **Data Mining: Practical Machine Learning Tools and Techniques**

- ✓ Auteur : Ian H. Witten & Eibe Frank
- ✓ Éditeur : Morgan Kaufmann Publishers
- ✓ Edition : Juin 2005 - 664 pages - ISBN : 0-12-088407-0

# Ressources

<https://www.technologynetworks.com/informatics/articles/calculating-a-least-squares-regression-line-equation-example-explanation-310265>

<https://machinelearningmastery.com/simple-linear-regression-tutorial-for-machine-learning/>

<https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931>

<https://machinelearningmastery.com/implement-simple-linear-regression-scratch-python/>