Fouille de Données

Data Mining

La Régression

Plan du cours

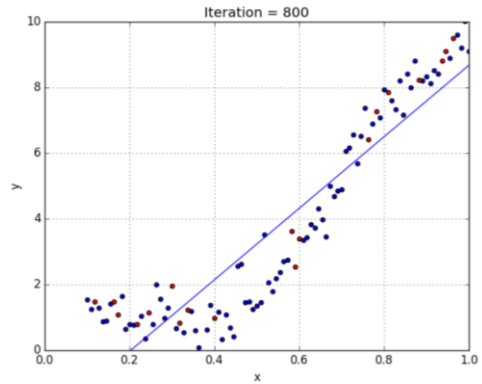
1. Régression / Estimation : Définition et principe
2. Régression linéaire simple.
3. Régression linéaire simple en utilisant le Gradient Descent.
4. Régression linéaire multiple

Classification : Algorithmes

SAVOIR - PREDIRE - DECIDER

Données

$$f(x)=b_1x+b_0$$



Connaissances

Linear Regression

La régression

- La **régression** est la méthode utilisée pour l'estimation des valeurs **continues**. Tache supervisée.

- 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 Xi (attributs). = Expliquer une variable *Y* à l'aide d'une variable *X* ou plusieurs variables *X*i.

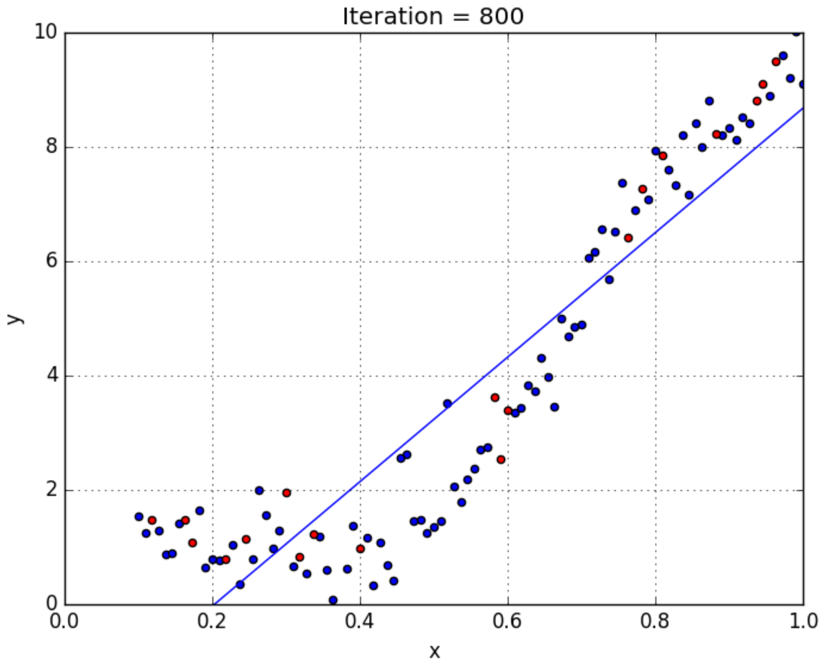
- Ex : prédire le cours de la bourse, le prix d'un appartement, ou bien l'évolution de la température sur Terre.

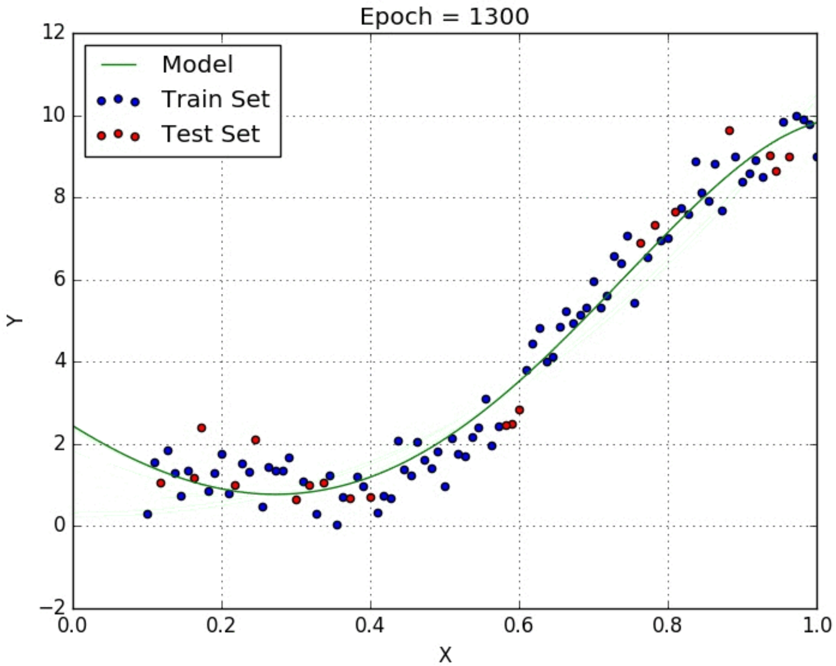
- Il s'agit donc de trouver une fonction f (=le modèle de régression) qui se rapproche le plus possible d'un scénario donné d'entrées et de sorties.

- Différents types de régression : linéaire, polynomiale, logistique (classification), Lasso, etc.

La régression

- Différents types de régression : linéaire, polynomiale.

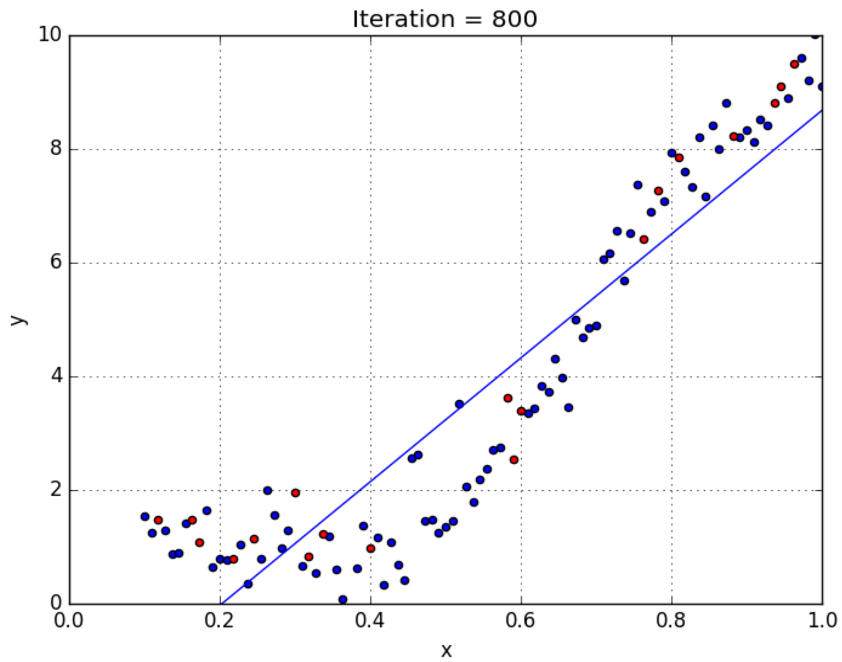


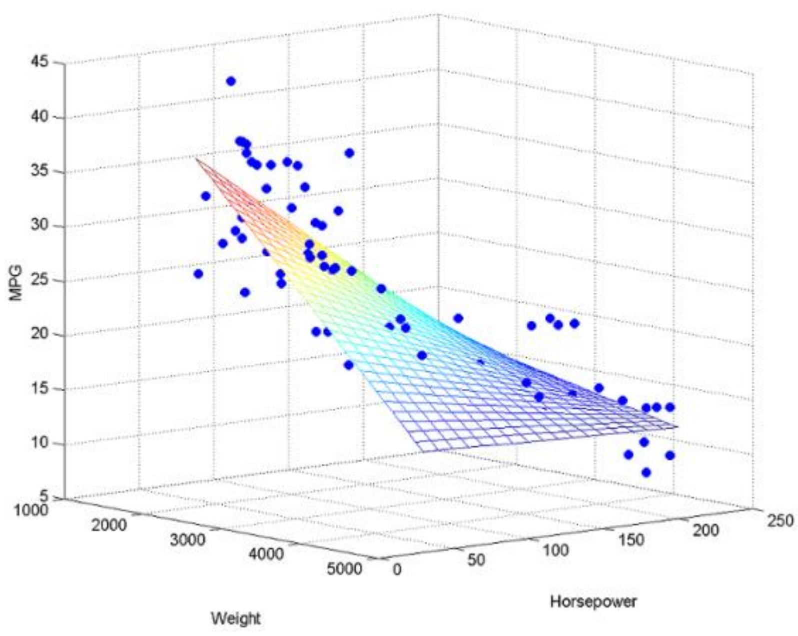


La régression

- 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 Xi.



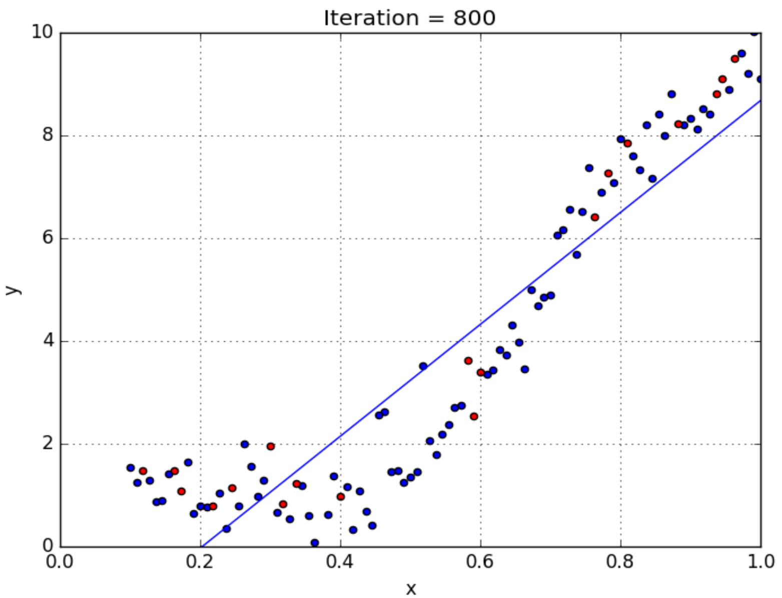


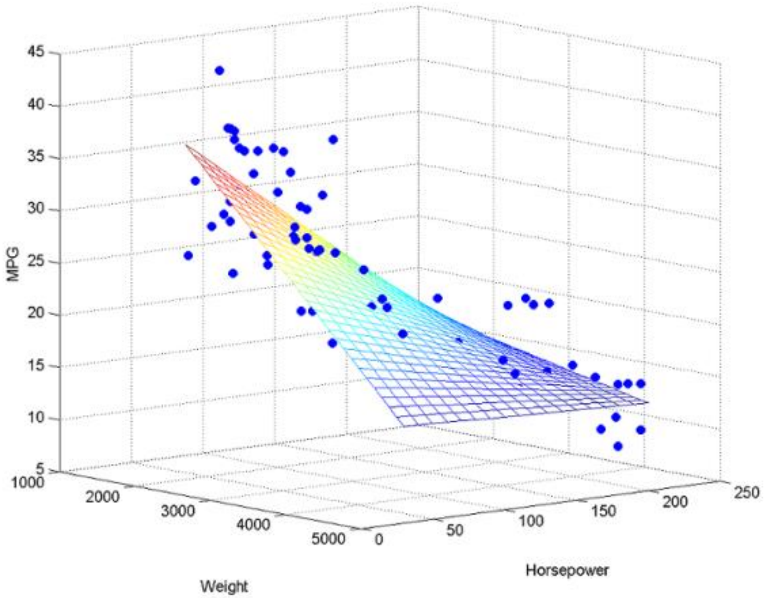
La régression

- 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_1) et du nombre de pièces (X_2) .

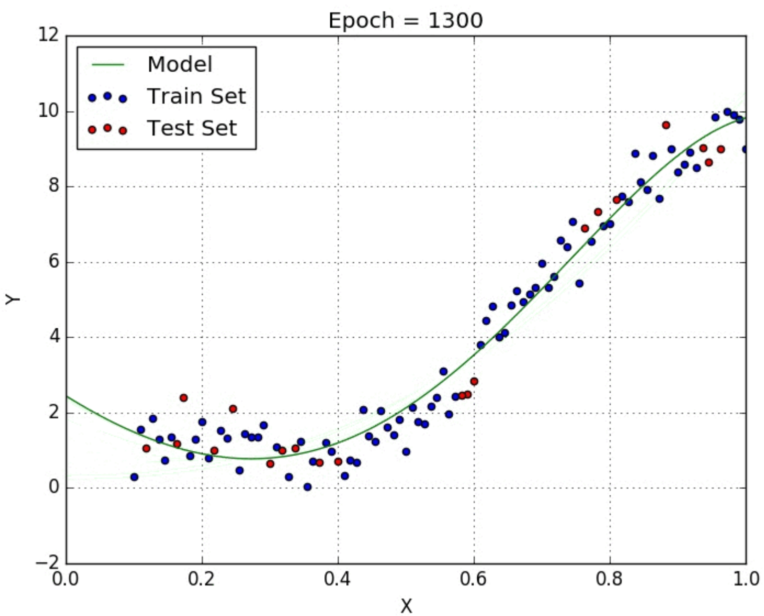


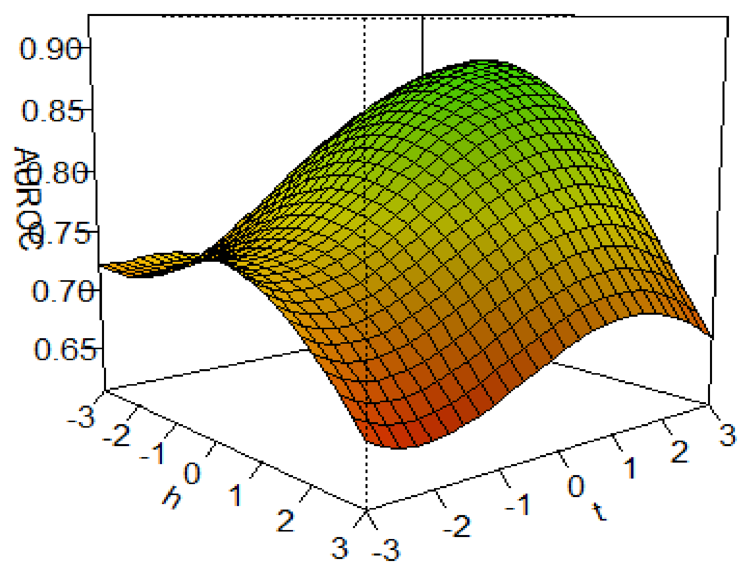


La régression

- 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 Xi.





La régression linéaire simple

Régression **linéaire simple**

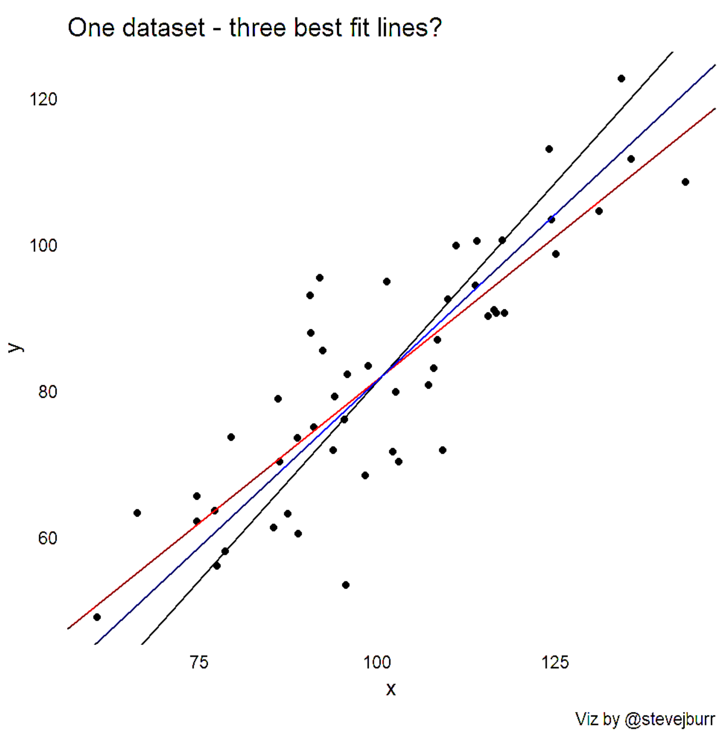
- **But** : Trouver un **modèle linéaire** $f(x) = b1x + b0$ où $b1$ et $b0$ sont les **hyperparamètres/coefficients** du modèle.

- **y = b1x + b0**

- Trouver le **meilleur** modèle (Best fit line) =>

=> Trouver les **meilleures** (optimales) valeurs des paramètres **b1 (slope)** et **b0 (intercept)**.

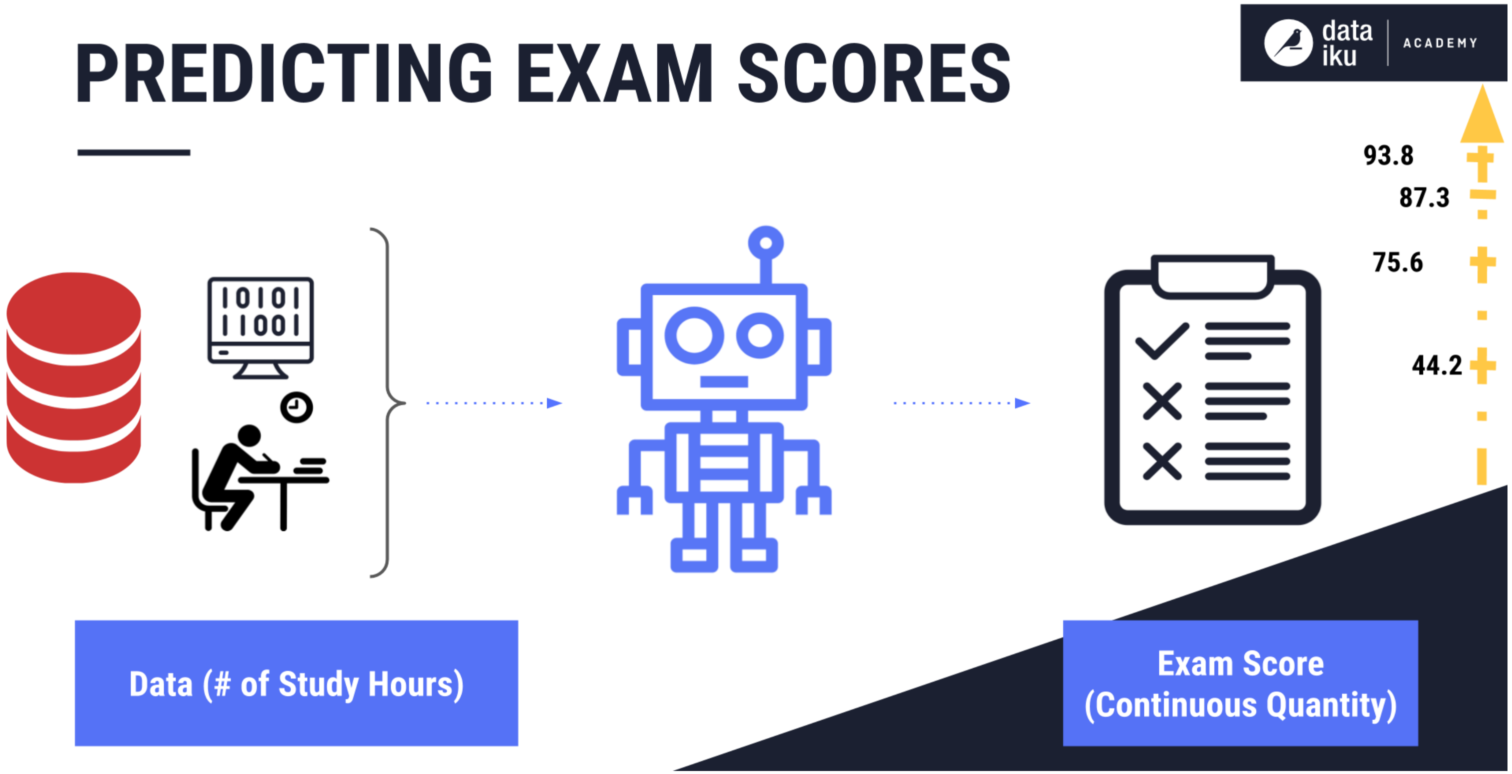
=> Faire le **minimum d'erreurs** possible sur les prédictions de Y.



La régression linéaire simple

Régression **linéaire simple**

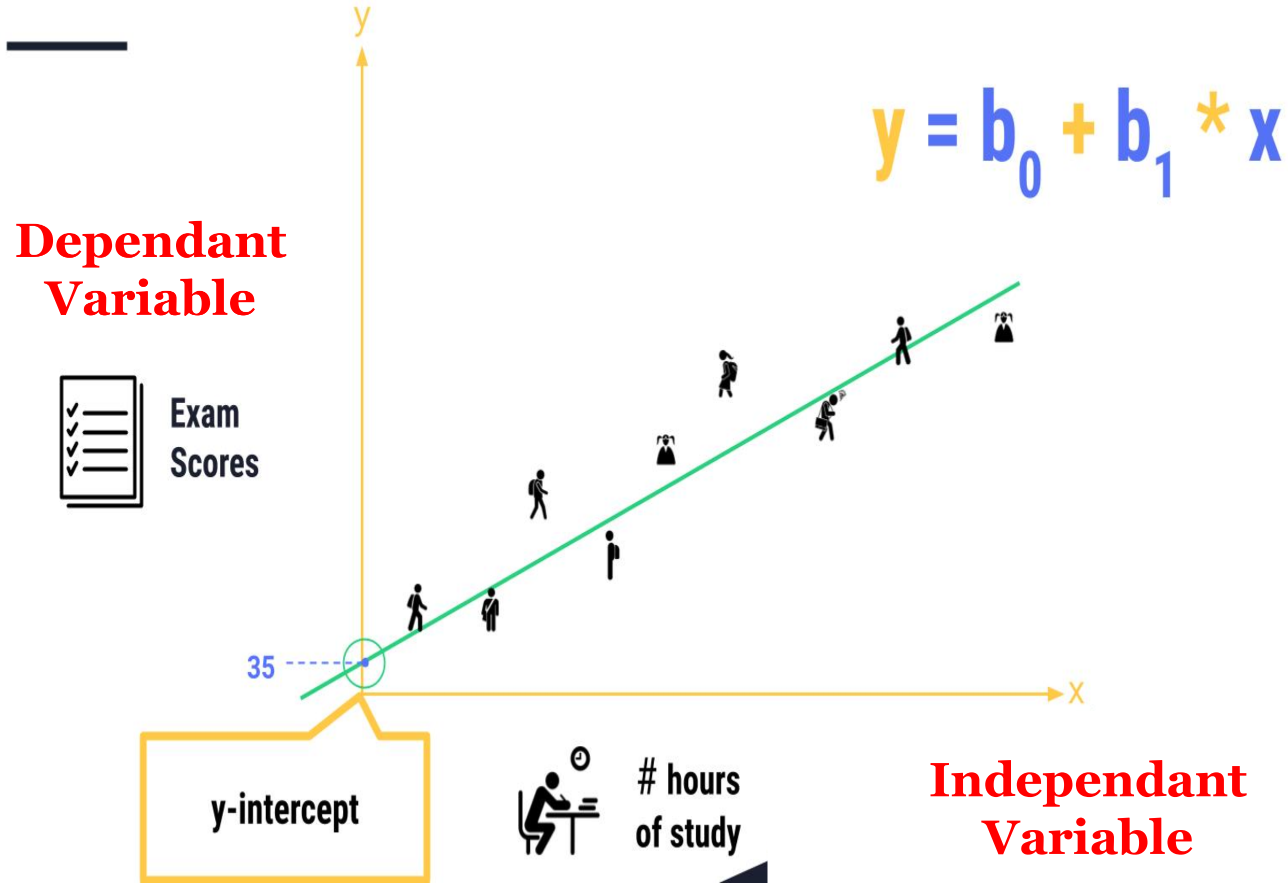
- **But** : Trouver un **modèle linéaire** $f(x) = b_1 x + b_0$
- Exemple :



La régression linéaire simple

Régression **linéaire simple**

- **But** : Trouver un **modèle linéaire** $y = b1x + b0$



La régression linéaire simple

Régression **linéaire simple**

- **Exemple :**

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

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

Objectif :

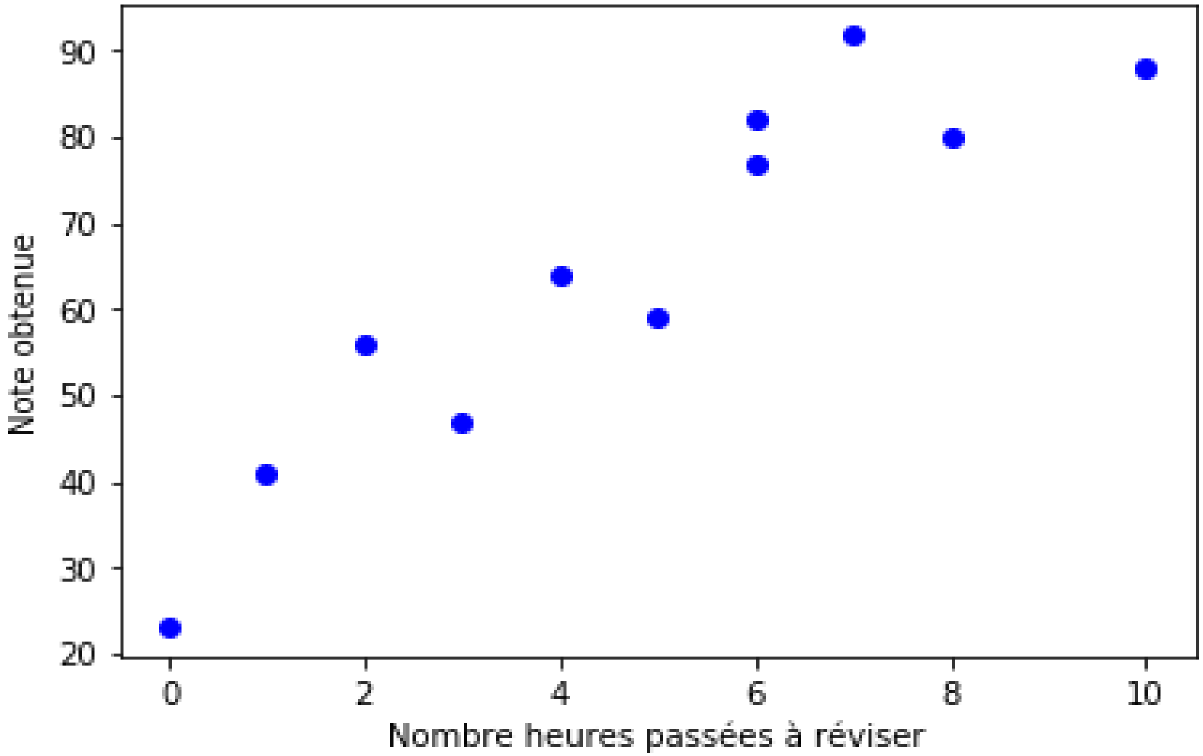
On souhaite savoir si, de façon générale, le nombre d'heures passées à réviser a une **influence** sur la note obtenue et sous quelle forme cette influence peut être exprimée.

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.

La régression linéaire simple

Régression **linéaire simple**

- **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**

- **Training - Entrainement :**

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

$$\begin{cases} b_1 = \dfrac{\sum(x-\bar{x}) * (y-\bar{y})}{\sum(x-\bar{x})^2} = \dfrac{Covariance(x,y)}{Variance(x)} \\ \\ b_0 = \bar{y} - b_1\bar{x} \end{cases}$$

La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Training - Entrainement :**

1 - Calculer les valeurs de **b1** (le slope) et **bo** (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
6	82
10	88
2	56
4	64
6	77
7	92
0	23
1	41
8	80
5	59
3	47

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

La régression linéaire simple

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

Hours spent on essay	Grade	Hours spent – Average Hours Spent $(x - \bar{x})$
6	82	1.27
10	88	5.27
2	56	-2.73
4	64	-0.73
6	77	1.27
7	92	2.27
0	23	-4.73
1	41	-3.73
8	80	3.27
5	59	0.27
3	47	-1.73

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

La régression linéaire simple

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 $(y - \bar{y})$
6	82	1.27	17.55
10	88	5.27	23.55
2	56	-2.73	-8.45
4	64	-0.73	-0.45
6	77	1.27	12.55
7	92	2.27	27.55
0	23	-4.73	-41.45
1	41	-3.73	-23.45
8	80	3.27	15.55
5	59	0.27	-5.45
3	47	-1.73	-17.45

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

La régression linéaire simple

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

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**

- **Training – Entrainement :**

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

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

∑(x-x¯) * (y-y¯) = 611.36

∑(x-x¯) ^2 = 94.18

$\rightarrow$ **b1 = 611.36/94.18**

= 6.49

La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Training – Entrainement :**

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

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

b1 = 6.49

→ **bo = 64.45 – (6.49 * 4,72)**

= 33.81

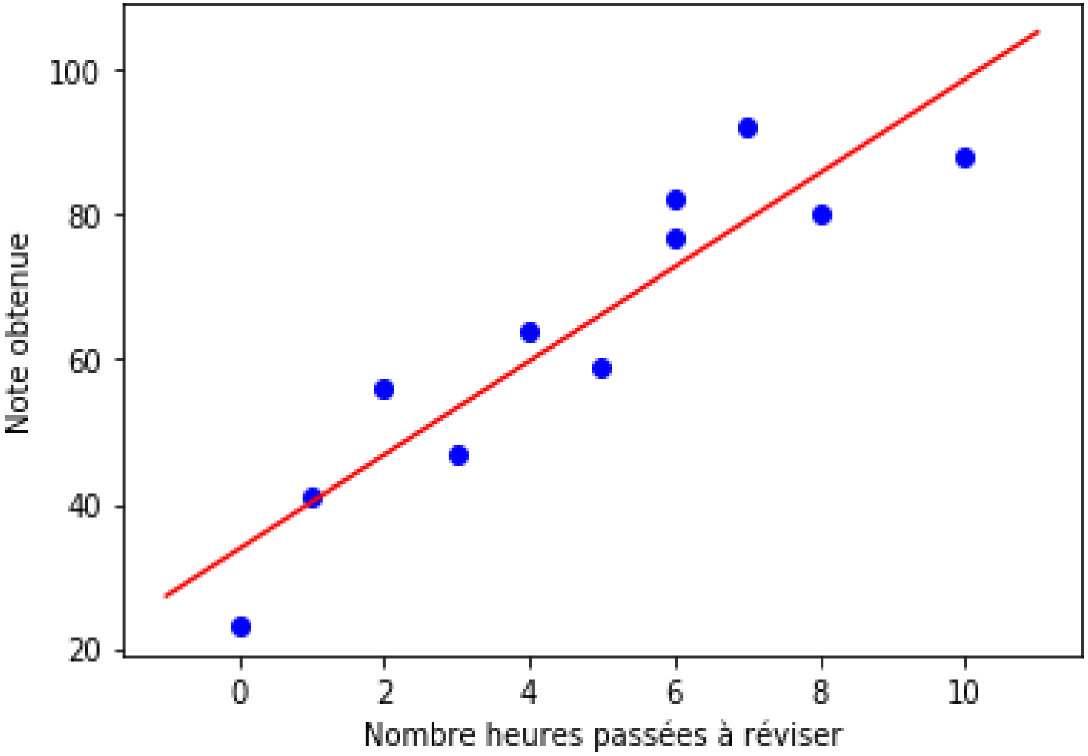
Mean	4.72	64.45
	$\bar{x}$	$\bar{y}$

La régression linéaire simple

<u>Régression linéaire simple – **Etapes de base**</u>

- **Training – Entrainement :**

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



La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Prédiction :** Prédire la valeur Y (la note) d'un nouvel exemple (**X=6**) selon la fonction trouvée précédemment:

Hours spent on essay	Grade
6	**?**

$$\mathbf{y = 6.49 * x + 33.81}$$

➔

$$\mathbf{Y} = \mathbf{6.49 * (6) + 33.81}$$

$$= \mathbf{72.71}$$

La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Evaluation du modèle :** Prédire les nouvelles valeurs Y des exemples de la **table d'entrainement**, selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

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

➔

Predicted Grade
72.716216
98.681467
46.750965
59.733591
72.716216
79.207529
33.768340
40.259653
85.698842
66.224903
53.242278

La régression linéaire simple

Régression linéaire simple – **Etapes de base**

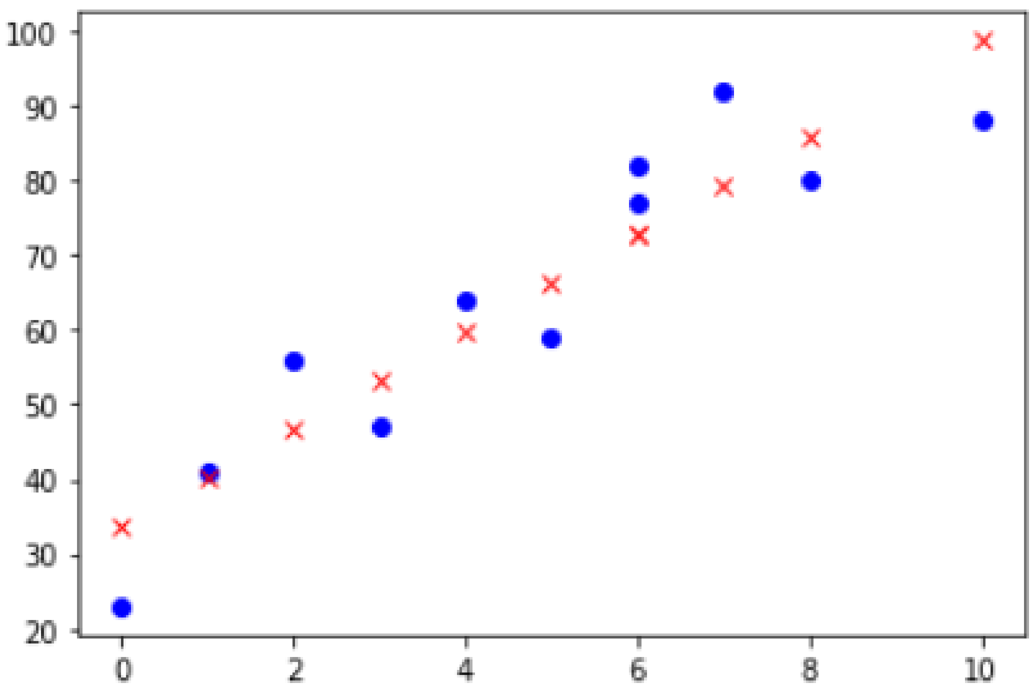
- **Evaluation du modèle :** Prédire les nouvelles valeurs Y des exemples de la **table d'entrainement**, selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

En bleu : vraie valeur - **y**
En rouge : valeur prédite – **y_pred**

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

Predicted Grade
72.716216
98.681467
46.750965
59.733591
72.716216
79.207529
33.768340
40.259653
85.698842
66.224903
53.242278



La régression linéaire simple

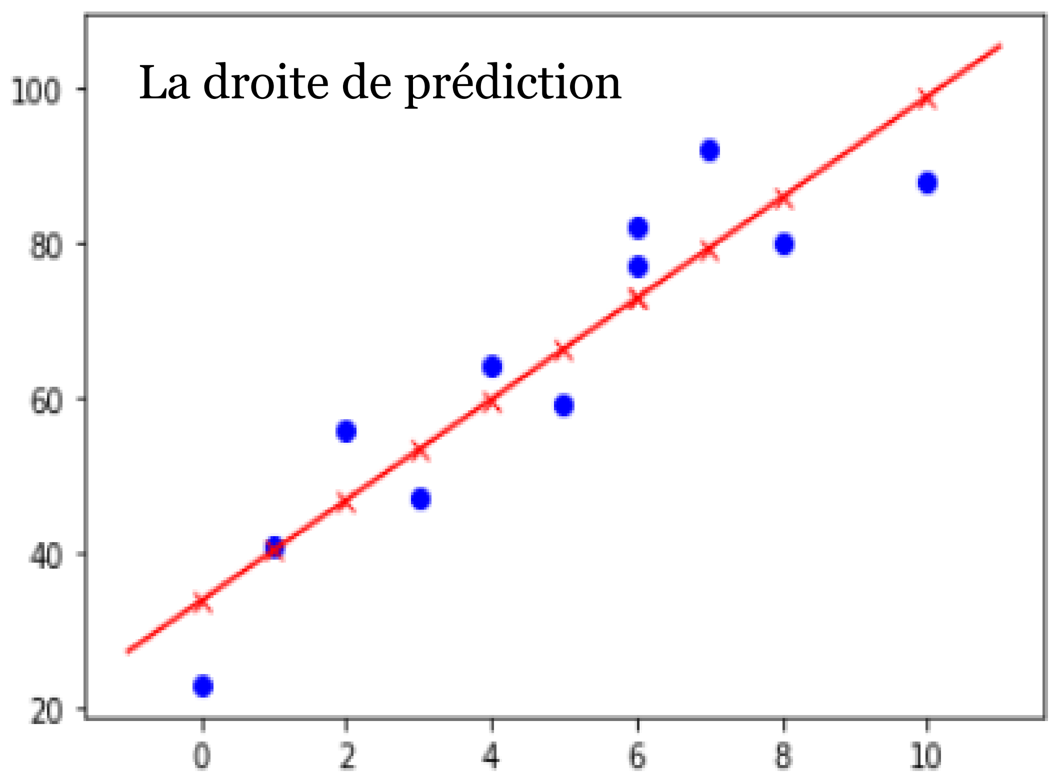
Régression linéaire simple – Etapes de base

- **Evaluation du modèle :** Prédire les nouvelles valeurs Y des exemples de la **table d'entrainement**, selon la fonction trouvée précédemment:

y = 6.49 * x + 33.81

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

Predicted Grade
72.716216
98.681467
46.750965
59.733591
72.716216
79.207529
33.768340
40.259653
85.698842
66.224903
53.242278



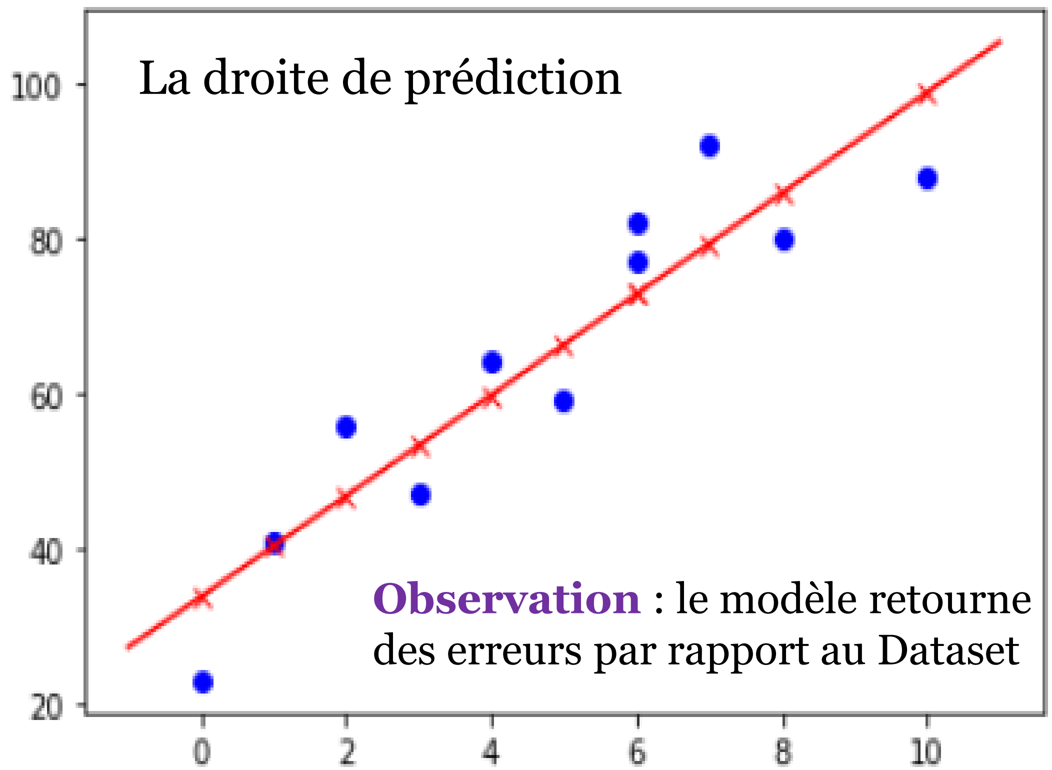
La régression linéaire simple

<u>Régression linéaire simple – **Etapes de base**</u>

- **Evaluation du modèle :** Prédire les nouvelles valeurs Y des exemples de la **table d'entrainement**, selon la fonction trouvée précédemment:

$$y = 6.49 * x + 33.81$$

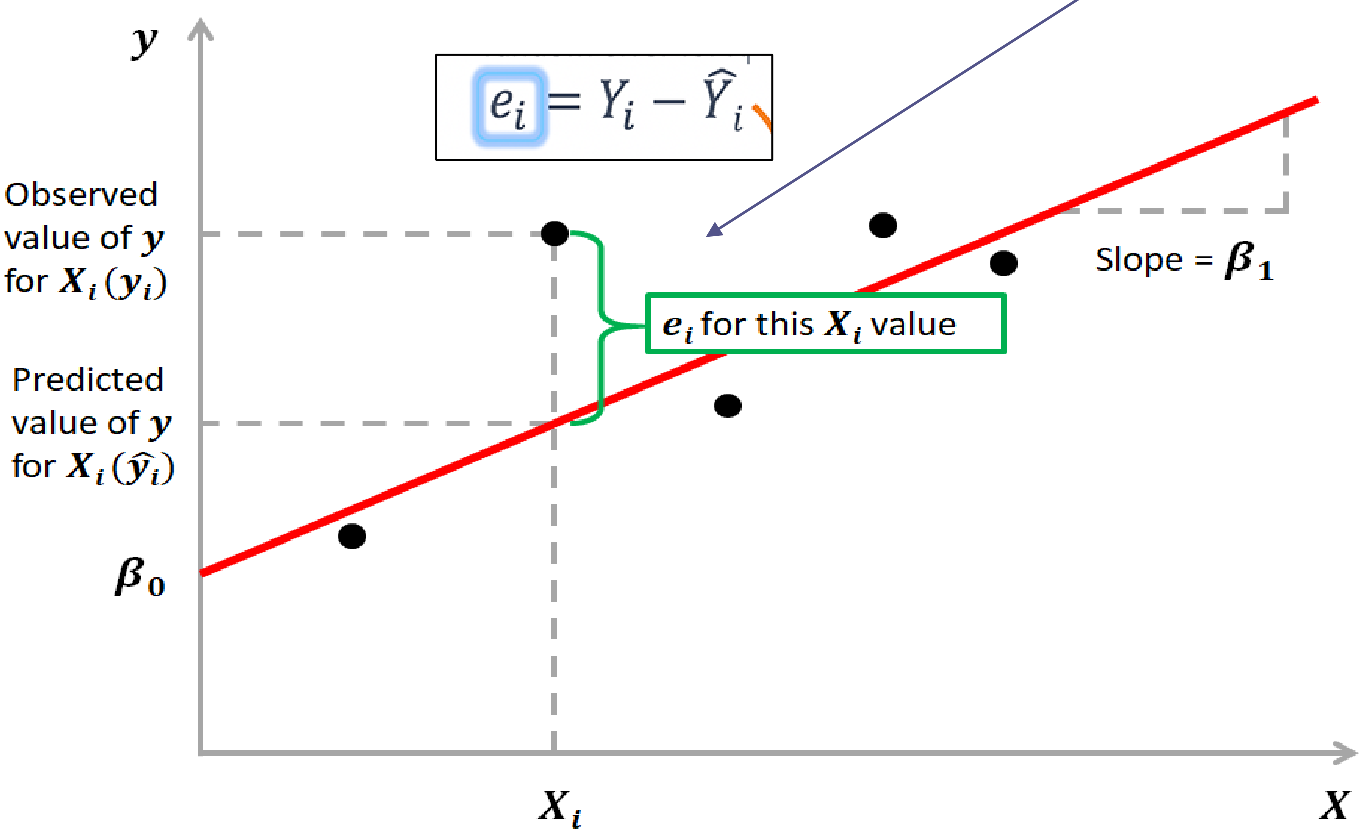
Hours spent on essay	Grade	Predicted Grade
6	82	72.716216
10	88	98.681467
2	56	46.750965
4	64	59.733591
6	77	72.716216
7	92	79.207529
0	23	33.768340
1	41	40.259653
8	80	85.698842
5	59	66.224903
3	47	53.242278



La régression linéaire simple

Régression linéaire simple – **Etapes de base**

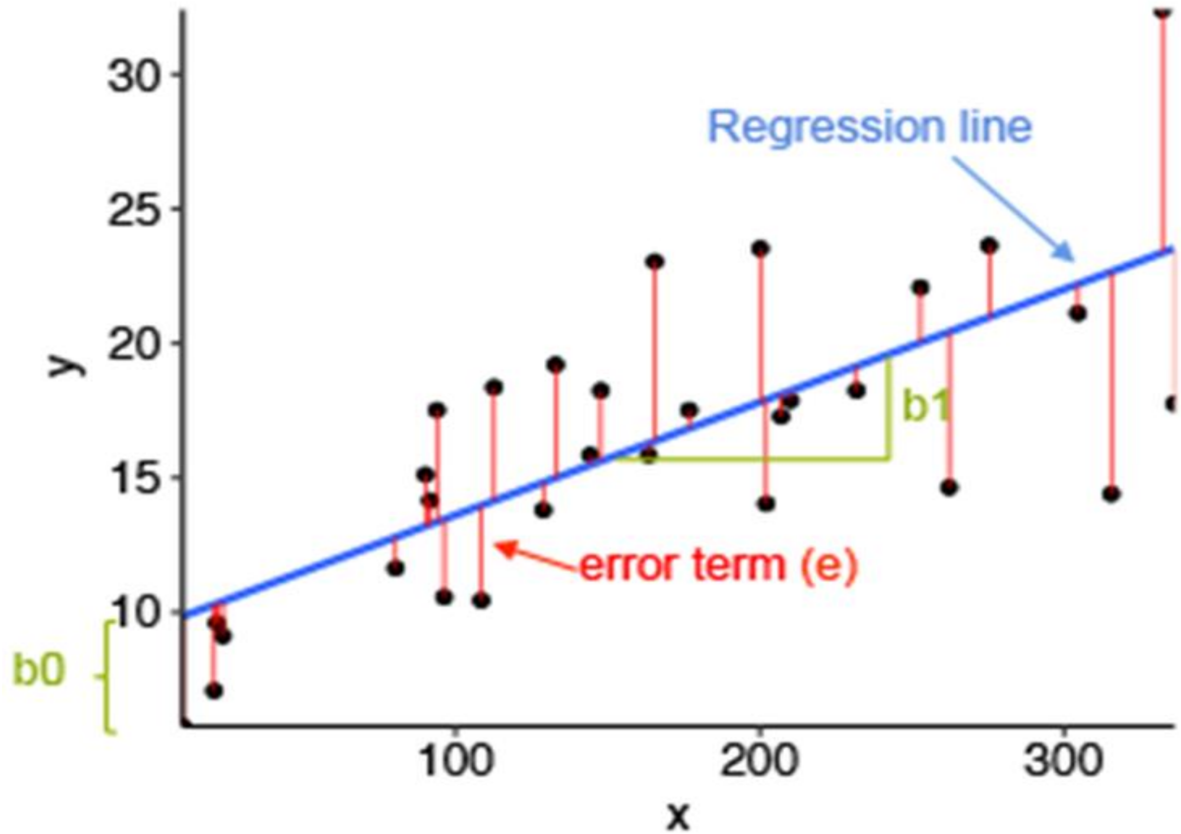
- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions y_pred (i.e. f(x)) :



La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions y_pred (i.e. f(x)) :



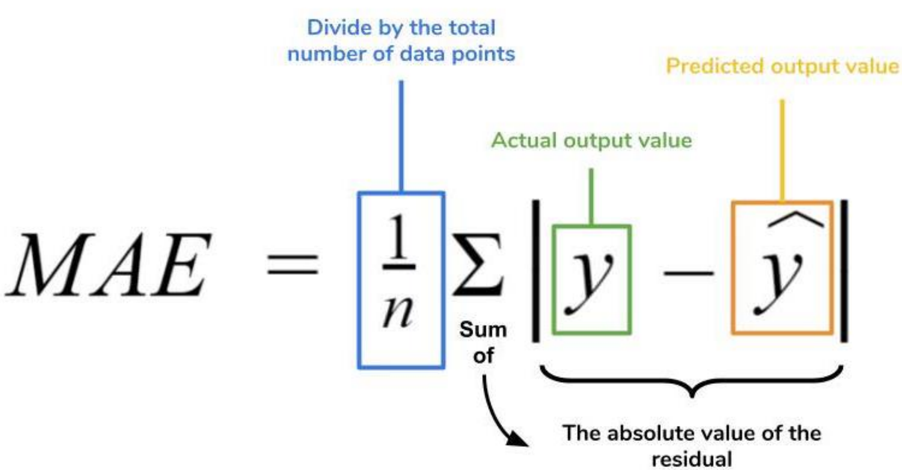
La régression linéaire simple

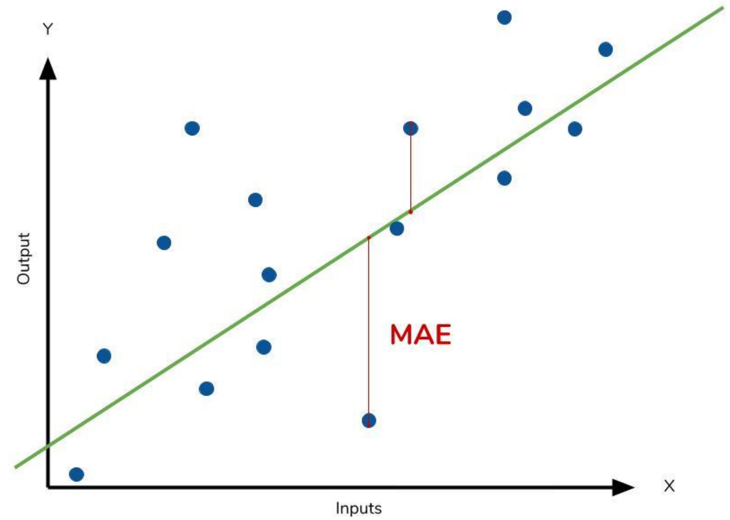
<u>Régression linéaire simple – **Etapes de base**</u>

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions y_pred (i.e. f(x)) :

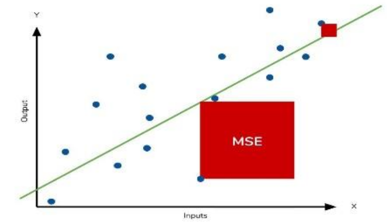
- Différentes **fonctions de coût** permettant d'estimer l'erreur d'un modèle.

- Ex: **MAE** – Mean Absolute Error / l'erreur absolue moyenne.





La régression linéaire simple



<u>Régression linéaire simple – **Etapes de base**</u>

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions y_pred (i.e. f(x)) :

- Différentes **fonctions de coût** permettant d'estimer l'erreur d'un modèle.

- Ex: **MSE** – Mean Squared Error / l'erreur quadratique moyenne.

- Ex: **RMSE** – Root Mean Squared Error / sqrt (l'erreur quadratique moyenne).

$$MSE = \frac{1}{n} \Sigma \underbrace{\left(y - \widehat{y} \right)^2}_{\text{The square of the difference between actual and predicted}}$$

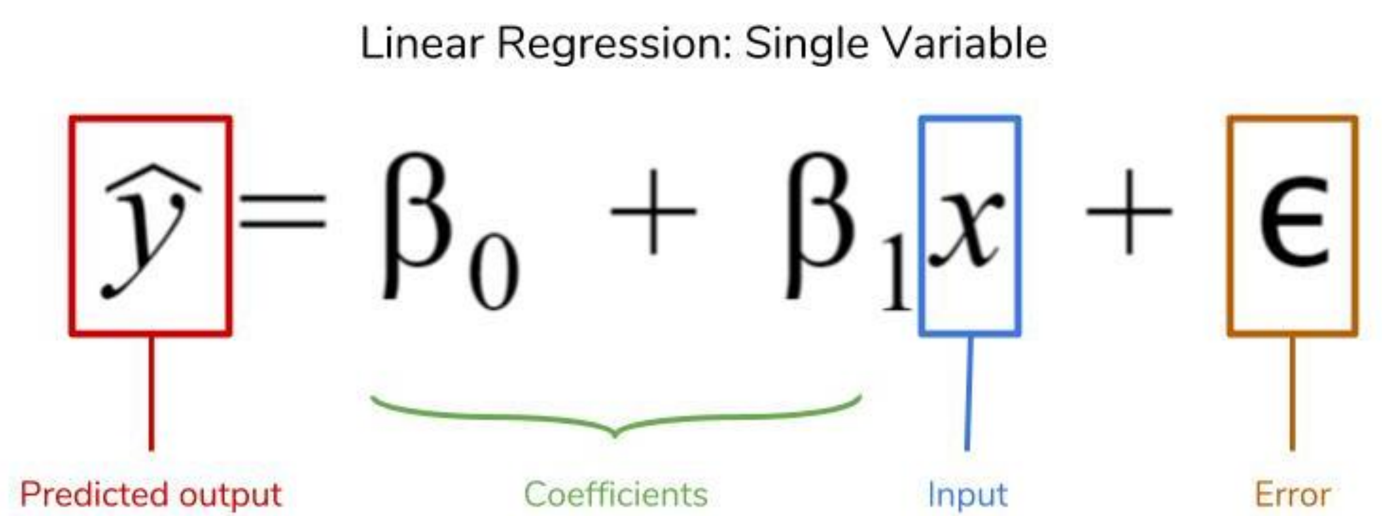
$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

La régression linéaire simple

<u>Régression linéaire simple – **Etapes de base**</u>

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions **y_pred** (i.e. f(x)) :

- **y_pred = b1x + b0 + err**



La régression linéaire simple

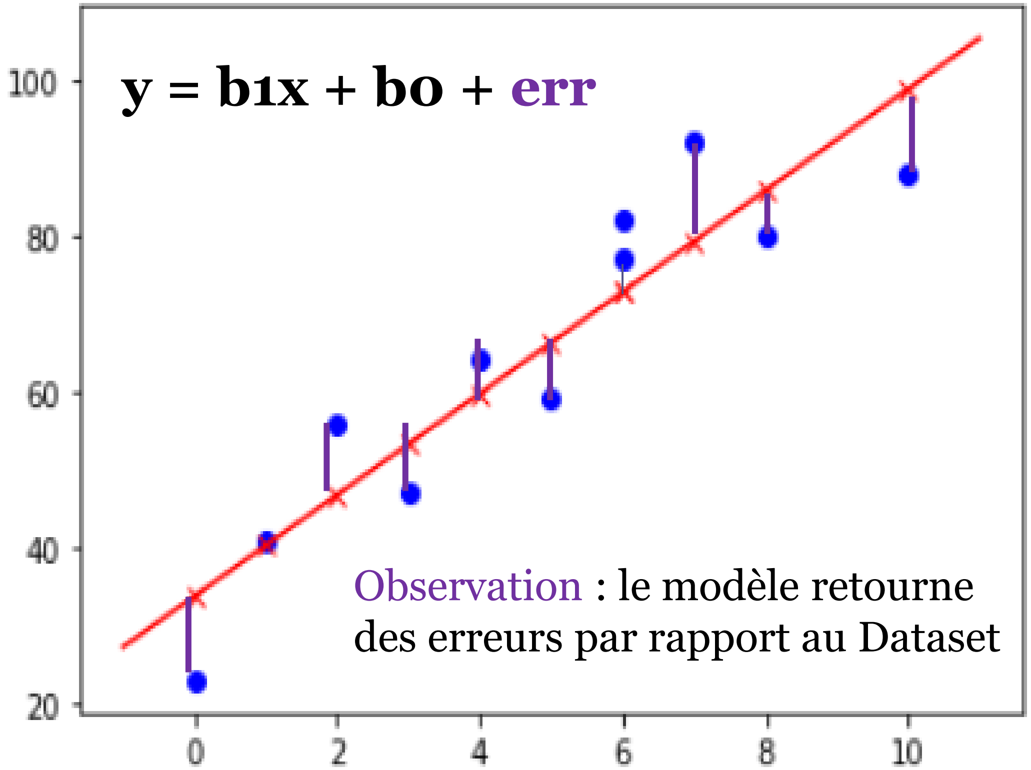
Régression linéaire simple – **Etapes de base**

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions **y_pred** (i.e. f(x)) :

$$\boxed{e_i} = Y_i - \widehat{Y}_i$$

Ex : (y_pred – y) = 85.698842 - 80 = +5.698842

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



La régression linéaire simple

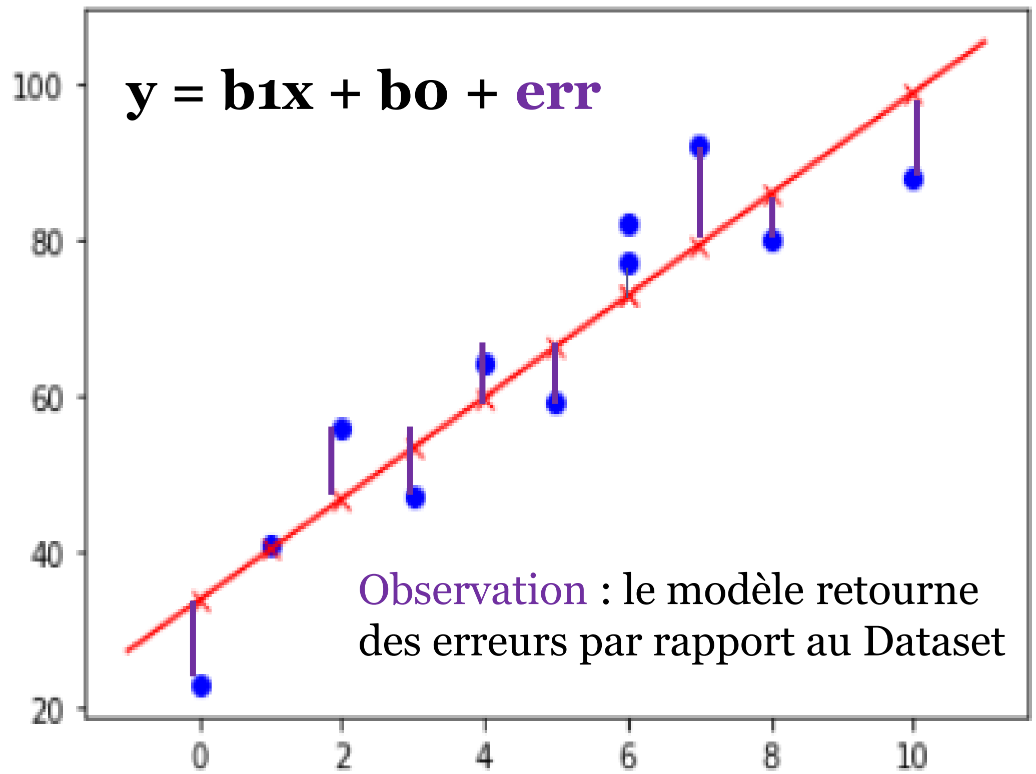
Régression linéaire simple – **Etapes de base**

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions **y_pred** (i.e. f(x)) :

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

=> **RMSE** = 8.125

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



La régression linéaire simple

Régression linéaire simple – **Etapes de base**

- **Evaluation du modèle** de régression en estimant les **erreurs** sur les prédictions **y_pred** (i.e. f(x)).
- Quel est **le meilleur modèle** ? Laquelle de ces droites est la droite la mieux ajustée (performante) ?

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

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

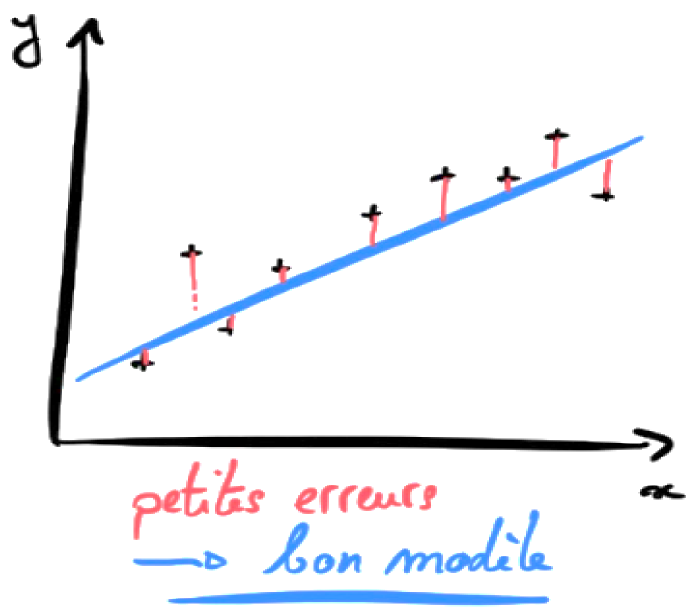
La régression linéaire simple

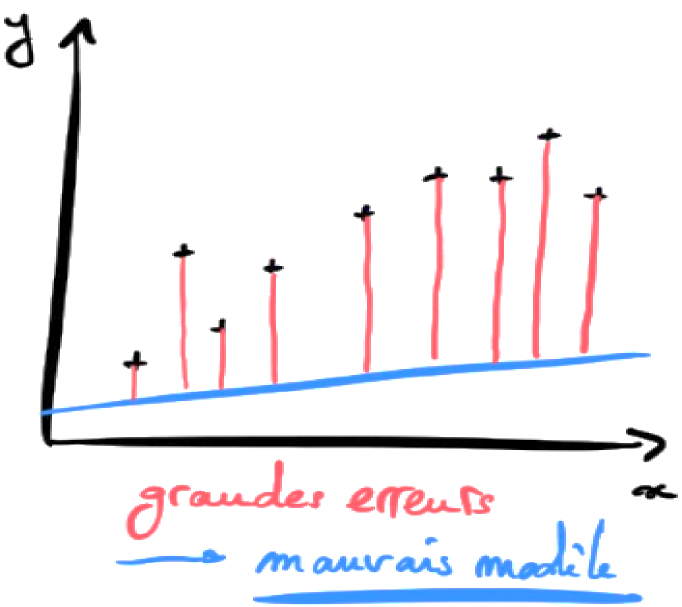
Régression linéaire simple – **Etapes de base**

- **Minimiser l'erreur** en minimisant la fonction coût:

- **y = b1x + b0 + err** => Ramener **err** vers **zéro** => RMSE le plus petit possible

- Le but est de trouver les meilleures (**optimales**) estimations des coefficients **b0** et **b1** pour minimiser les erreurs de prédiction de y.





La régression linéaire simple

Régression linéaire simple – Etapes de base

- **Minimiser l'erreur** en minimisant la fonction coût:

- **y = b1x + b0 + err** => Ramener **err** vers **zéro** => RMSE le plus petit possible

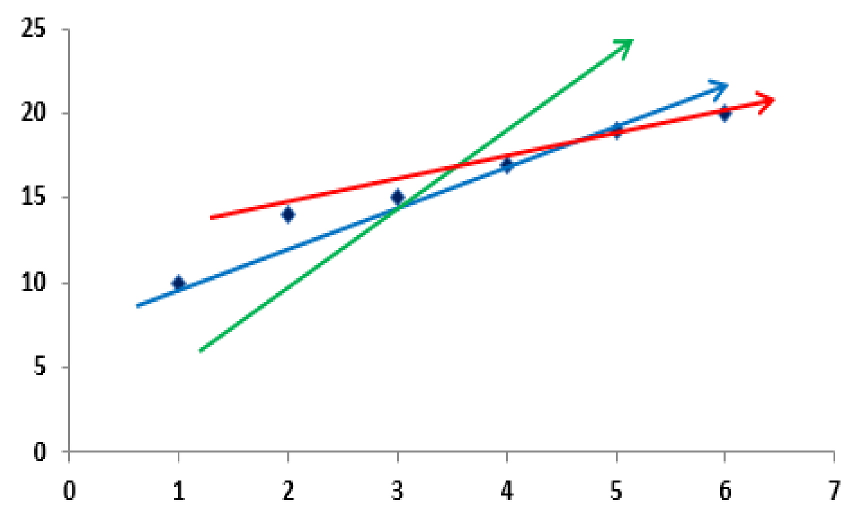
- Le but est de trouver les meilleures (**optimales**) estimations des coefficients **b0** et **b1** pour minimiser les erreurs de prédiction de y.

which of these lines is the best fit line?

Minimiser l'erreur ➔

Problème d'optimisation ➔

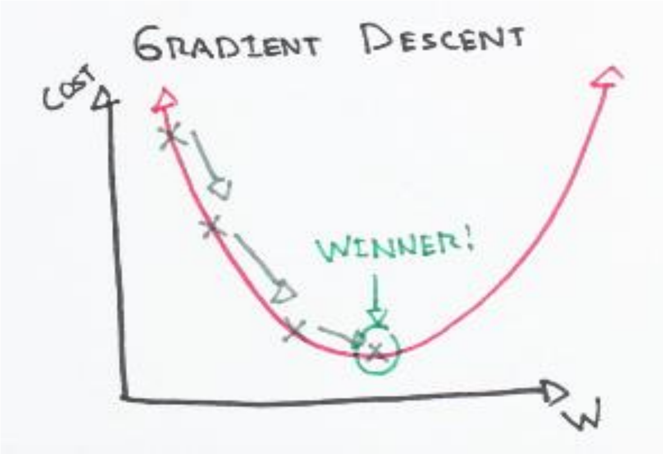
Itérer les étapes précédentes en utilisant un **algorithme d'optimisation** cherchant à minimiser la **fonction coût**.

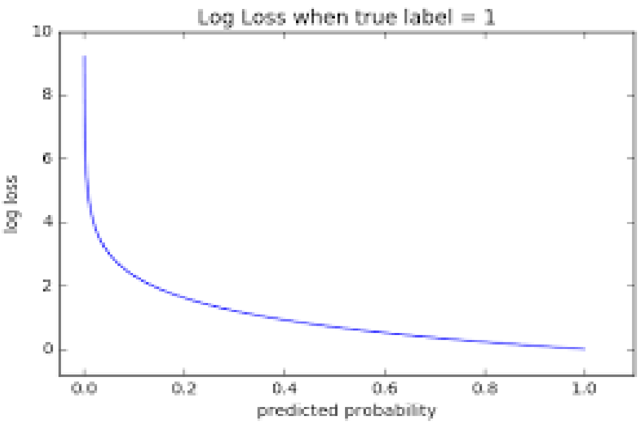


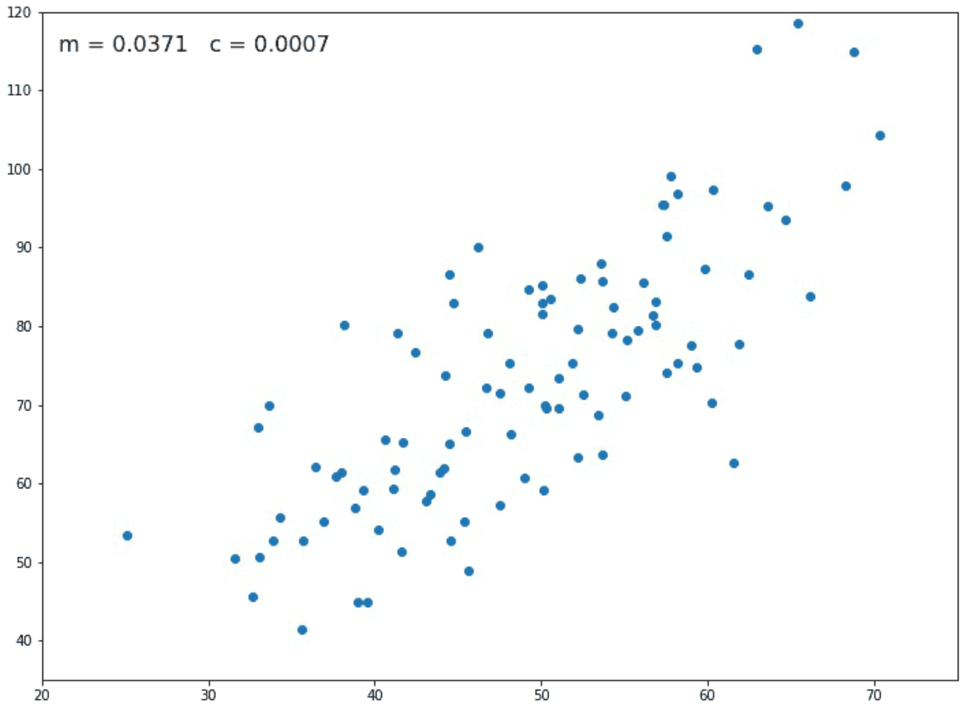
La régression linéaire simple

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

Ex : Algorithme de Gradient Descent (Descente du Gradient)





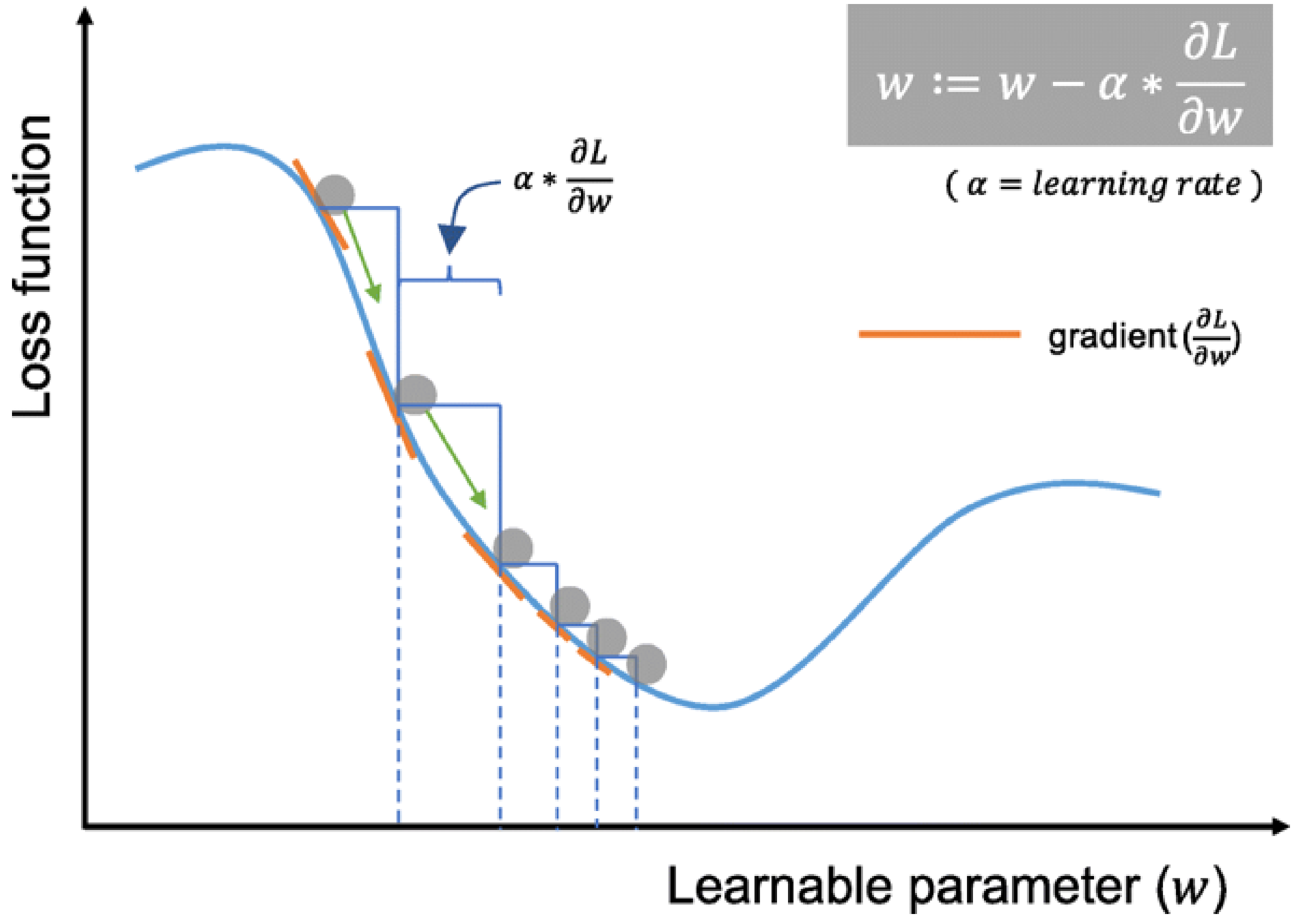


Gif : https://miro.medium.com/max/1400/1*CjTBNFUEI_IokEOXJ0ozKw.gif

La régression linéaire simple

<u>Régression linéaire simple – **Utilisation d'un algorithme d'optimisation**</u>

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



b0 and b1

La régression linéaire simple

<u>Régression linéaire simple – **Utilisation d'un algorithme d'optimisation**</u>

Ex : Algorithme de Gradient Descent (Descente du Gradient)

***<u>Etapes</u>* :**

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

- Fonction cout : **MSE** – Mean Squared Error

y_vrai y_pred

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

1 - Initialiser les paramètres ***m*** et **c** à 0 : m =**0** et c = **0** (m est b1 et c est b0)

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

<u>Régression linéaire simple – **Utilisation d'un algorithme d'optimisation**</u>

Ex : Algorithme de Gradient Descent (Descente du Gradient)

***<u>Etapes</u>* :**

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

- Fonction cout : **MSE** – Mean Squared Error

y_vrai y_pred

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

1 - Initialiser les paramètres ***m*** et **c** à 0 : m =**0** et c = **0** (m est b1 et c est b0)

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**

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 ***Dm*** et ***Dc*** :

- **Dm** : Dérivée partielle de la fonction coût selon le paramètre m
- **Dc** : Dérivée partielle de la fonction coût selon le paramètre c

$$Cost\ Function(MSE) = \frac{1}{n}\sum_{i=0}^{n}(y_i - y_{i\,pred})^2$$

Replace $y_{i\,pred}$ with $mx_i + c$

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

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

$$\boxed{= \frac{-2}{n}\sum_{i=0}^{n} x_i\left(y_i - y_{i\,pred}\right)}$$

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 ***Dm*** et ***Dc*** :

- **Dm** : Dérivée partielle de la fonction coût selon le paramètre m
- **Dc** : Dérivée partielle de la fonction coût selon le paramètre c

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

$$= \frac{-2}{n}\sum_{i=0}^{n} x_i(y_i - y_{i\,pred})$$

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

La régression linéaire simple

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

Ex : Algorithme de Gradient Descent (Descente du Gradient)

***Etapes* :**

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

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

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

La régression linéaire simple

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

Ex : Algorithme de Gradient Descent (Descente du Gradient)

***Etapes* :**

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).

La régression linéaire simple

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

Ex : Algorithme de Gradient Descent (Descente du Gradient)

***Etapes* :**

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

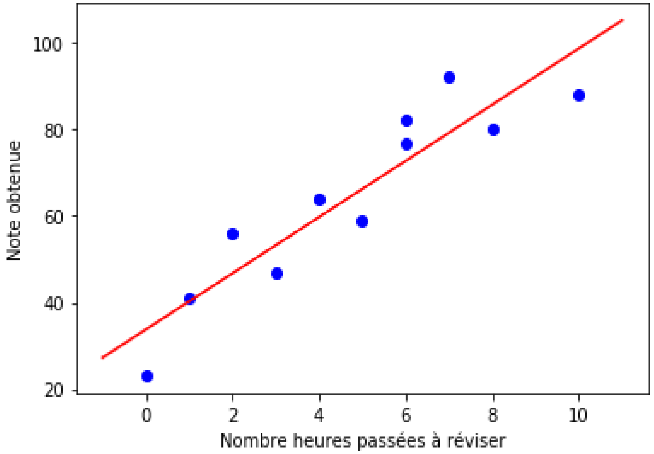
Régression linéaire simple – **Utilisation d'un algorithme d'optimisation**

Ex : Algorithme de Gradient Descent (Descente du Gradient)

Entrainement et Exécution sur 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
```

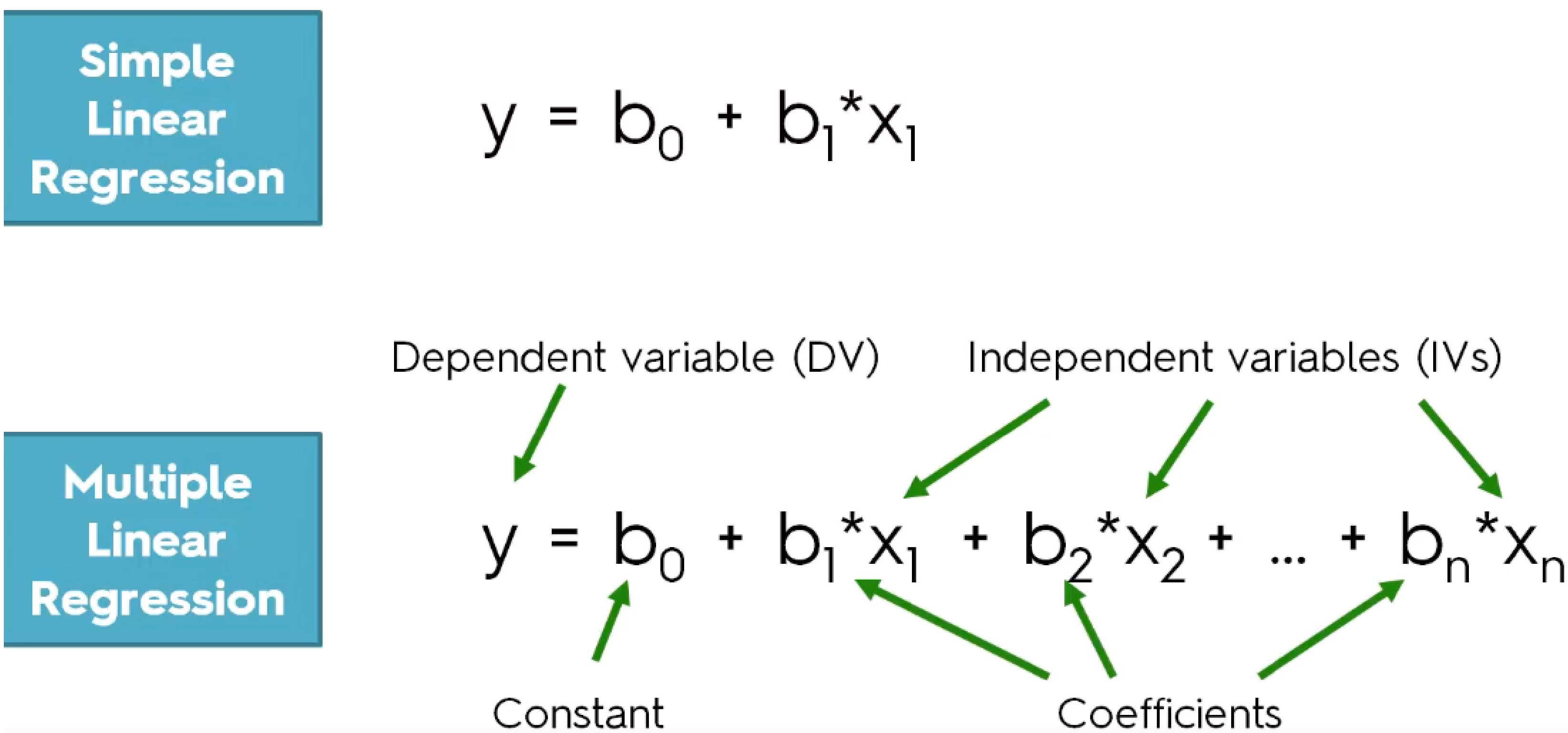


f(x) : y = m * x + c

La régression linéaire multiple

<u>Régression **linéaire multiple** :</u>

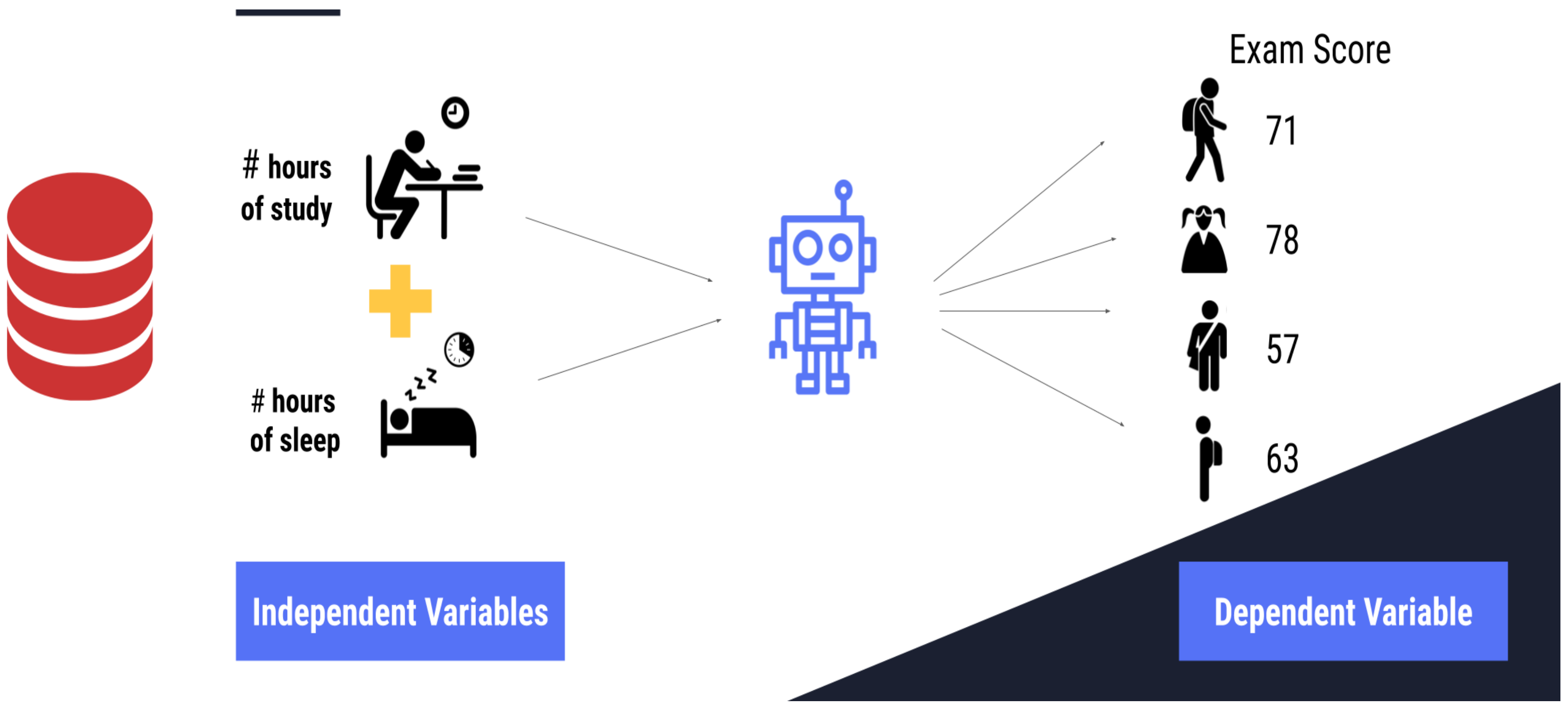
- La régression linéaire multiple utilisée pour estimer une sortie en fonction de **plusieurs entrées X** n'est qu'une extension de la précédente:



La régression linéaire multiple

Régression **linéaire multiple** : $$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_n X_{ni}$$

- La régression linéaire multiple utilisée pour estimer une sortie en fonction de **plusieurs entrées X** n'est qu'une extension de la précédente. Ex:



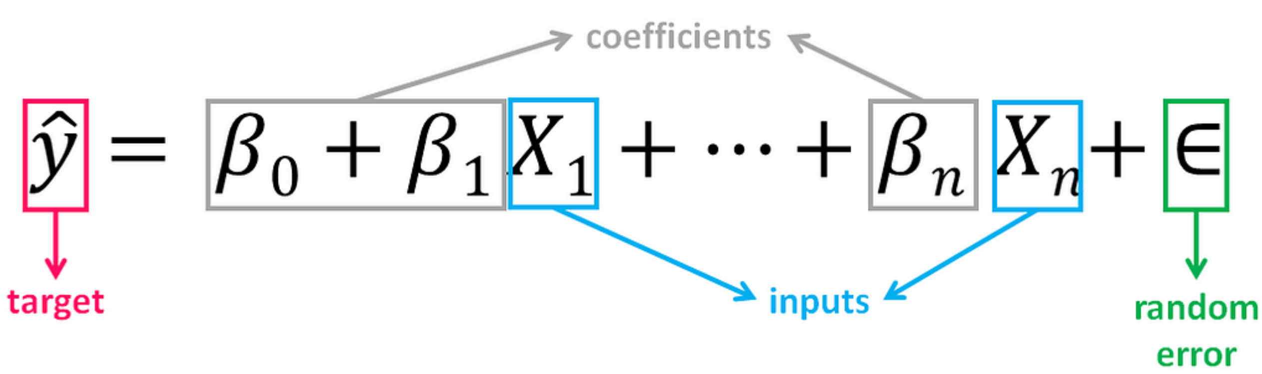
La régression linéaire multiple

<u>Régression **linéaire multiple** :</u> $Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_n X_{ni}$.

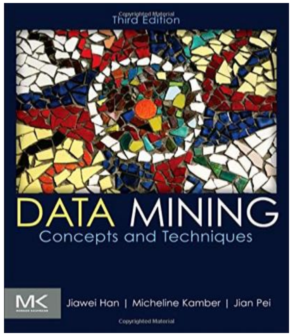
- La régression linéaire multiple utilisée pour estimer une sortie en fonction de **plusieurs entrées X** n'est qu'une extension de la précédente. Ex:

$$sales = \beta_0 + \beta_1 TV + \beta_2\, radio + \beta_3\, newspaper$$

- Calcul et optimisation des paramètres du modèle : b0, b1, b2, …. , bn.

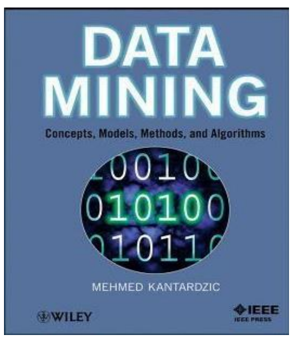


Références



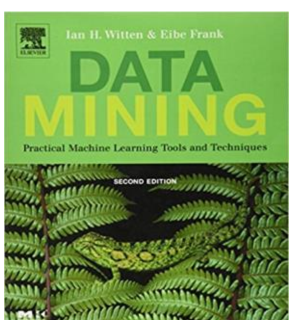
Data Mining : concepts and techniques, 3rd Edition

- ✓ 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

Références

- 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/