Apprentissage supervisé

Introduction à la Data Science - Efreitech

Maxime Jumelle - Taieb Badis

www.blent.ai

Plan

Fondements théoriques

Principe

En apprentissage supervisé, on dispose :

- des observations $\mathbf{X} = (X_i)_{1 \leq i \leq n}$ à valeurs dans $\mathbb{R}^p$
- des *labels* observés $\mathbf{y} = (y_i)_{1 \leq i \leq n}$ à valeurs dans $\mathbb{R}$

L'objectif est de trouver une fonction $f : \mathbb{R}^d \to \mathbb{R}$ qui approxime au mieux les labels (y_i) à partir des données (X_i) :

$$f(\mathbf{X}) = \hat{\mathbf{y}} \approx \mathbf{y}$$

Autrement dit, le modèle va *apprendre* selon les observations $\mathbf{X}$ pour prédire de nouvelles observations inconnues.

Comment apprendre ?

Cette notion d'apprentissage n'est pas évidente à mettre en pratique.

Pour cela, il nous faut un critère que l'on cherche à minimiser, tel que l'erreur commise entre les labels observés et les labels prédits.

C'est pourquoi on introduit une **fonction de perte** (*loss function*) L qui "mesure" cet écart. Plus la valeur de L est basse, mieux le modèle approxime les labels théoriques.

Exemple de pertes

Perte quadratique :

$$L(x, y) = (x - y)^2$$

Perte 0/1 :

$$L(x, y) = \mathbf{1}_{\{x=y\}}$$

Perte *hinge* :

$$L(x, y) = \max(0, 1 - xy)$$

Perte exponentielle :

$$L(x, y) = \exp(-xy)$$

Le problème d'optimisation

Synthétiquement, le problème d'optimisation que tous les algorithmes d'apprentissage supervisé cherchent à résoudre est :

$$\min_{f \in \mathcal{C}} \mathbb{E}[L(f(\mathbf{X}), \mathbf{y})]$$

On dispose d'un ensemble $\mathcal{C}$ de prédicteurs "candidats", et on cherche à savoir quel est celui qui minimise la fonction de perte.

Problème $\Rightarrow$ La quantité à minimiser est aléatoire et $\mathcal{C}$ est grand, très très grand ...

Solution $\Rightarrow$ Restreindre $\mathcal{C}$ à un choix limité d'une certaine classe de modèles

Modèle linéaire

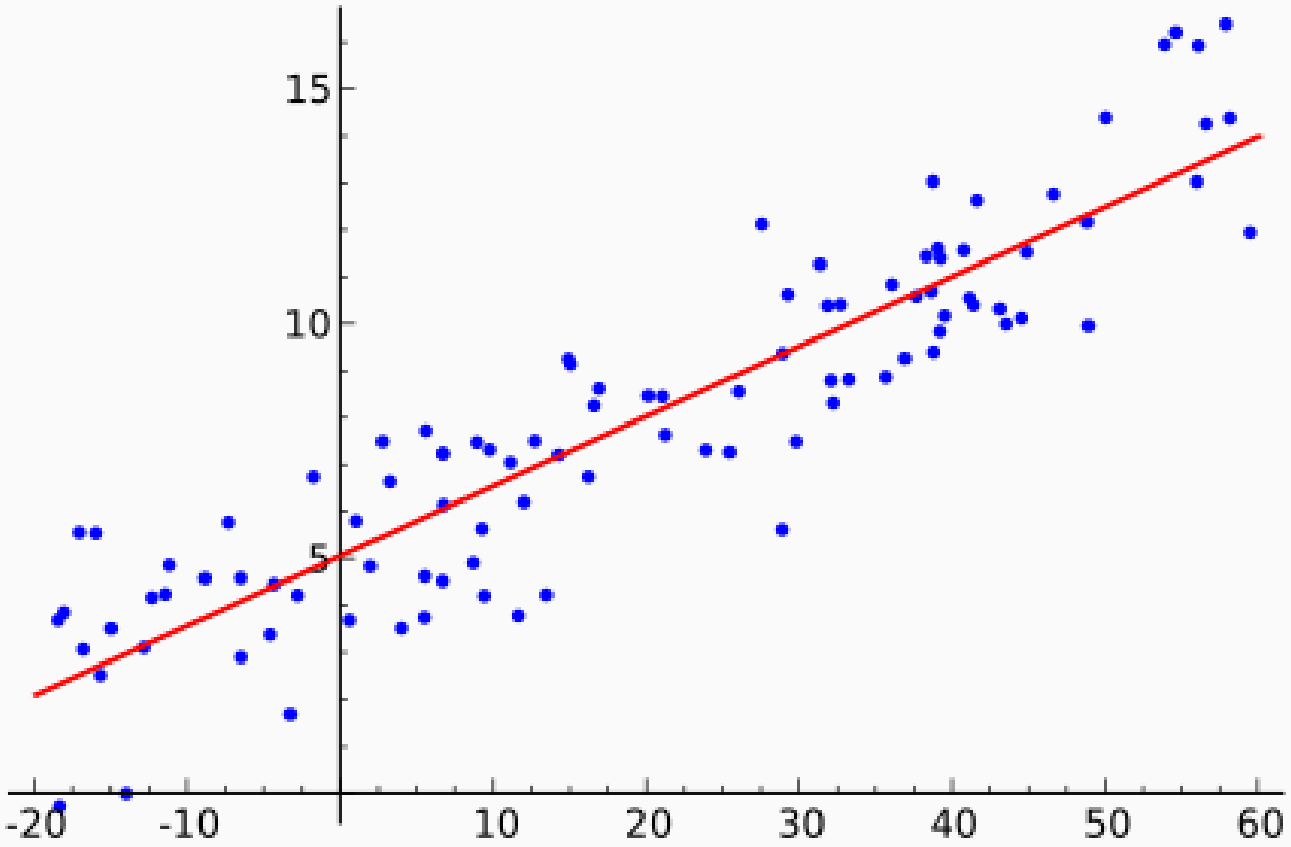
Fonctions affines

Considérons la cas classique d'une fonction affine :

$$y = ax + b$$

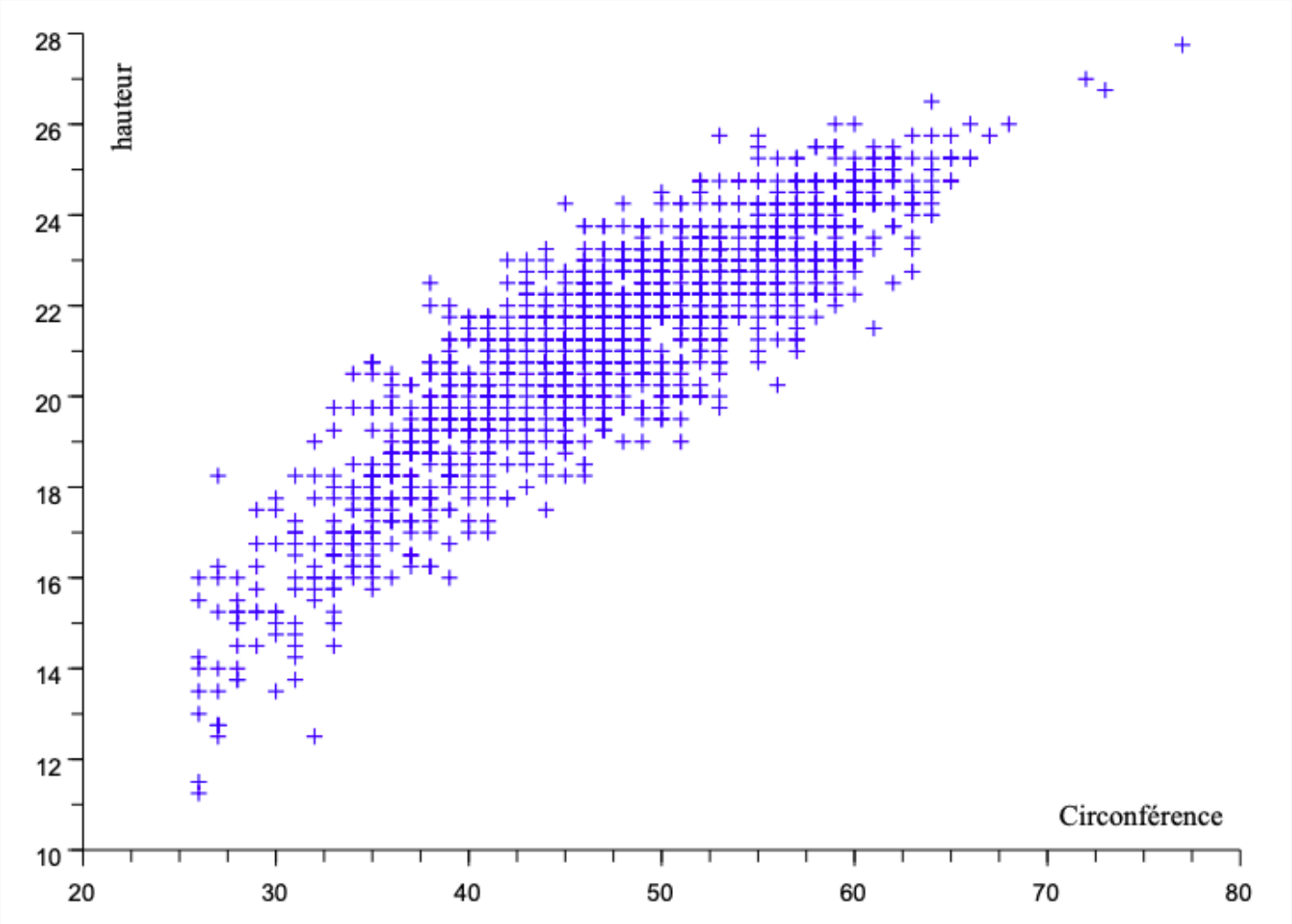
Ici, a et b sont des réels. Ces deux nombres définissent entièrement la courbe et permet donc d'obtenir une relation **affine** entre x et y. En statistique, cette relation est à la base des modèles dits **linéaires**, où une variable réponse se définit comme une somme de variables explicatives où chacune de ces dernières sont multipliés par un coefficient.

Fonctions affines



Exemple

Hauteur y (en m) d'eucalyptus en fonction de leur circonférence x (en m).



Modèle linéaire simple

Dans le modèle linéaire simple (une seule variable explicative), on suppose que la variable réponse suit le modèle suivant :

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

On remarque la ressemblance avec la fonction affine présentée ci-dessus. La différence réside dans l'existence du terme aléatoire (appelé bruit) ε_i. Afin de considérer le modèle, il est nécessaire de se placer sous les hypothèses suivantes.

$$(\mathcal{H}) : \begin{cases} \mathbb{E}[\varepsilon_i] = 0 \\ \mathrm{Cov}(\varepsilon_i, \varepsilon_j) = \delta_{ij}\sigma^2 \end{cases}$$

Modèle linéaire simple

Les différents éléments qui interviennent sont :

- β_0 : l'ordonnée à l'origine (nommée *intercept*)
- β_1 : le coefficient directeur
- x_i : l'observation i
- y_i : le i-ème label observé
- ε_i : le bruit aléatoire lié à la i-ème observation

La solution peut se calculer facilement via les formules fermées suivantes :

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

Exercice

Calculer $\hat{\beta}_0$ et $\hat{\beta}_1$ dans le cas où $\bar{x} = 0$ et $\bar{y} = 0$.

Estimateurs

Attention

$\hat{\beta}_0$ et $\hat{\beta}_1$ sont des **estimateurs** de β_0 et β_1 : on ne pourra jamais connaître la valeur des coefficients, on pourra seulement fournir des estimateurs !

- $\hat{\beta}_0$ et $\hat{\beta}_1$: variables aléatoires
- β_0 et β_1 : paramètres du modèle

On défini ainsi les labels prédits $\hat{y}_1, \ldots, \hat{y}_n$ par

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

Bruits et résidus

Comment comparer les labels observés et prédits ? Dans le modèle, nous avons vu la présence de bruit aléatoire ε_i. Néanmoins, pour prédire une observation, ce bruit n'intervient pas. Comment pouvons-nous ainsi estimer ce bruit ?

$$\text{Résidus : } \hat{\varepsilon}_i = y_i - \hat{y}_i$$

Ces résidus permettent d'estimer la variance des bruits, et donc de pouvoir les caractériser :

$$\hat{\sigma}^2 = \frac{\|\hat{\varepsilon}\|^2}{n-2} = \frac{1}{n-2}\sum_{i=1}^{n} \hat{\varepsilon}_i^2$$

Objectifs

Le modèle linéaire simple doit apprendre sur les données afin de déterminer des estimateurs $\hat{\beta}_0$ et $\hat{\beta}_1$ en minimisant la fonction de perte. Très souvent en régression, on minimise la perte quadratique moyenne (RMSE en anglais).

$$RMSE(Y, \hat{Y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} = \frac{1}{\sqrt{n}}\|Y - \hat{Y}\|_2$$

La RMSE mesure l'écart global entre les prédictions et les observations. Plus la RMSE est proche de 0, mieux le modèle prédit correctement les données conformément aux observations.

Exercice

Montrer que

$$RMSE^2(Y, \hat{Y}) = \frac{\|\hat{\varepsilon}\|^2}{n}$$

Score

Pour le modèle linéaire simple, une étude visuelle nous permet d'établir plus ou moins la performance du modèle.

En utilisant la définition des résidus, on peut définir le coefficient de détermination $R^2 \in [0, 1]$.

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

- R^2 proche de 1 : fort pouvoir explicatif
- R^2 proche de 0 : faible pouvoir explicatif

Modèle linéaire multiple

Dans le cas multiple (pour p variables explicatives), pour la i-ème observation, le modèle s'écrit :

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i$$

Ainsi, une observation x_i n'est plus une valeur, mais un **vecteur** $(x_{i1}, \ldots, x_{ip})$. Il est plus commode de regrouper les labels observés y_i et ces vecteurs d'observations x_i dans des matrices :

$$Y = X\beta + \varepsilon$$

Sous les hypothèses équivalentes du modèle simple en plus grande dimension

$$(\mathcal{H}) : \begin{cases} \text{rank}(X) = p \\ \mathbb{E}[\varepsilon] = 0 \text{ et } \text{Var}(\varepsilon) = \sigma^2 I_p \end{cases}$$

Modèle linéaire multiple

Les différents éléments qui interviennent sont :

- β : le vecteur directeur
- X : la matrice des observations
- Y : le vecteur des labels observés
- ε : le vecteur de bruit

Avec $X = (\mathbf{1}, X_1, \ldots, X_n)$, $Y = (y_1, \ldots, y_n)^\top$ et $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)^\top$. La solution des MCO (Moindres Carrés Ordinaires) est alors :

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y$$

Bruits et résidus

Les résidus s'obtiennent par le même calcul que pour la régression linéaire simple.

$$\text{Résidus : } \hat{\varepsilon}_i = y_i - \hat{y}_i$$

En revanche, l'estimateur de la variance dépends cette fois-ci du nombre de variables p :

$$\hat{\sigma}^2 = \frac{\|\hat{\varepsilon}\|^2}{n-p} = \frac{1}{n-p}\sum_{i=1}^{n} \hat{\varepsilon}_i^2$$

Score

Pour généraliser la notion de coefficient de détermination R^2 à une plus grande dimension, on introduit les quantités suivantes :

- $SCT = \|Y - \bar{y}\mathbf{1}\|^2$ est la somme des carrés totaux
- $SCE = \|\hat{Y} - \bar{y}\mathbf{1}\|^2$ est la somme des carrés expliqués
- $SCR = \|\hat{\varepsilon}\|^2$ est la somme des carrés résiduels

$$R^2 = 1 - \frac{SCR}{SCT} = \frac{SCE}{SCT}$$

Exercice

Retrouver la formule du R^2 dans le cas de la régression simple à partir de la formule ci-dessus.

TP

TP Jupyter : Modèle linéaire

Arbres de régression et de classification

Arbres

Les arbres sont utilisés pour des problèmes de régression ou de classification.

Il existe plusieurs méthodes pour construire un arbre. Nous nous focaliserons sur un des algorithmes les plus utilisés : CART.

CART est un algorithme développé par Leo Breiman, dérivé de ID3, à la différence où chaque noeud possède soit exactement deux noeuds enfants, soit aucun enfant (noeud terminal).

Détails

On souhaite expliquer une variable réponse Y en fonction de p variables sur n observations $X_1, \ldots, X_n$.

La variable réponse Y peut être soit discrète (arbre de classification), soit continue (arbre de régression), d'où l'appellation de l'algorithme CART.

Le modèle f que l'on cherche à construire est donc un arbre de classification ou de régression.

Classifieurs faibles

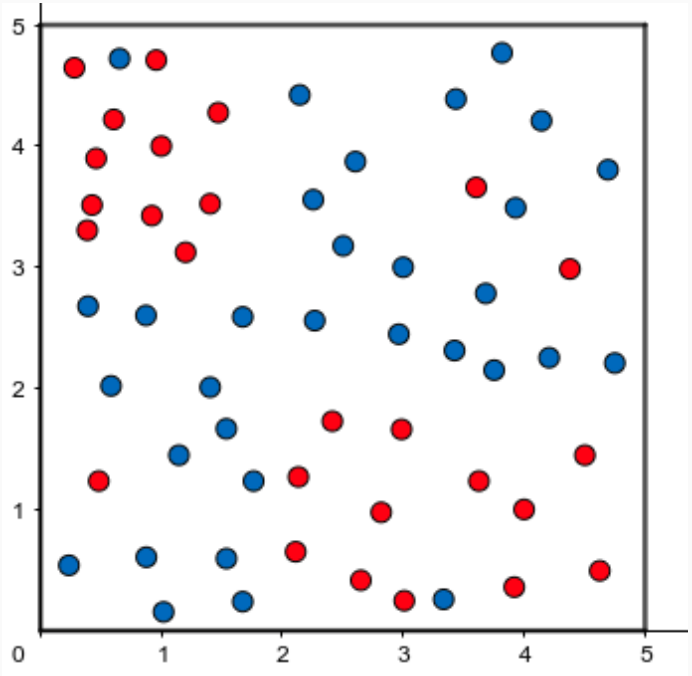
L'objectif d'un arbre est de déterminer une succession de classifieurs "faibles", c'est-à-dire de sous-modèles basiques.

La succession de ces classifieurs "faibles" a pour objectif de former un classifieur "complexe", capable de s'adapter efficacement à des données ayant beaucoup de variables.

D'une manière générale, on regroupe toutes ces méthodes dans un ensemble de techniques appelé **ensemble learning**.

Exemple pas-à-pas

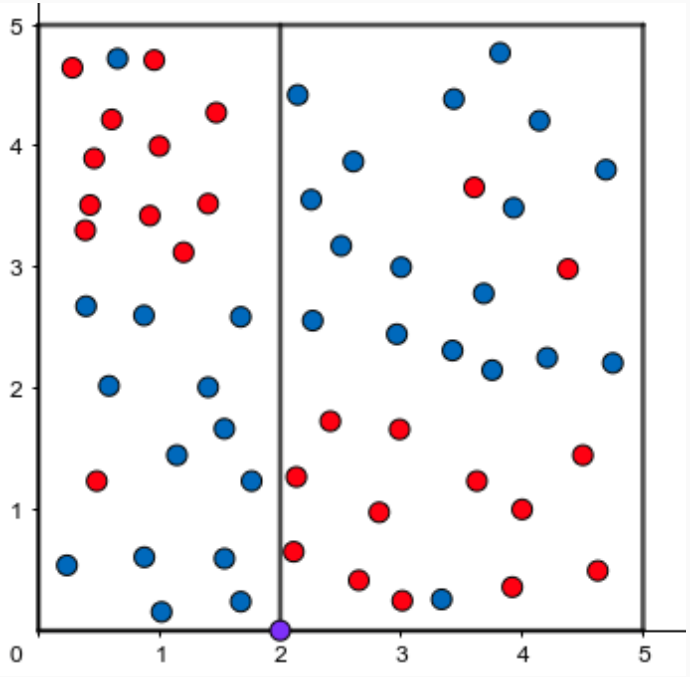
Détaillons un exemple de classification binaire pas-à-pas. On dispose :

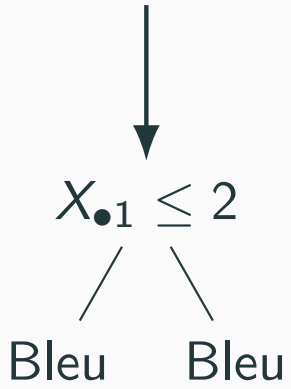


- de n observations $(X_{i1}, X_{i2})_{1 \leq i \leq n} \in \mathbb{R}^2$
- des labels $(Y_i)_{1 \leq i \leq n} \in \{\text{Rouge}, \text{Bleu}\}$

Pour décider de la valeur de sortie du noeud, on utilise un *majority vote* pour les problèmes de classification.

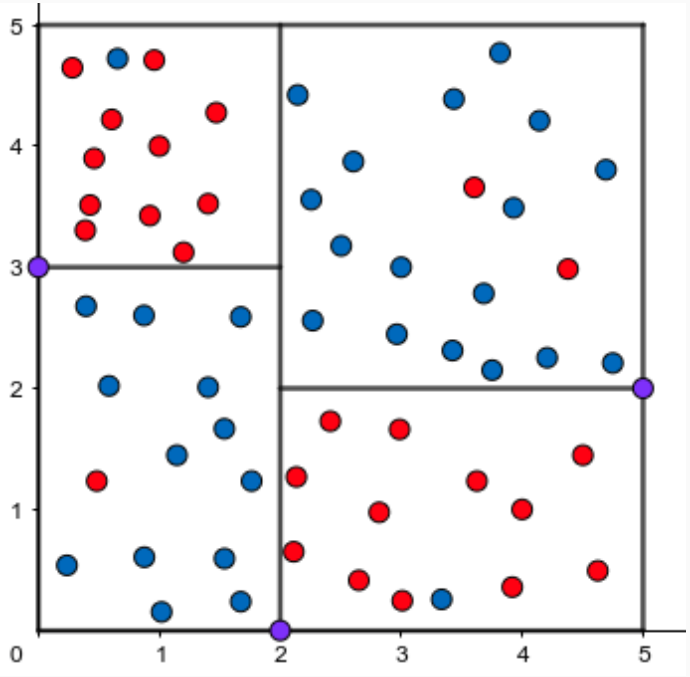
Exemple pas-à-pas

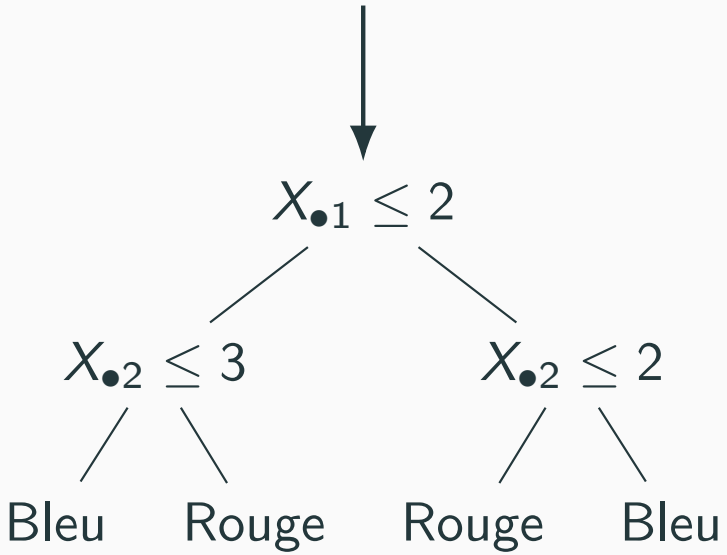




L'arbre n'est pas assez précis avec une profondeur de 1.

Exemple pas-à-pas





La précision devient satisfaisante dès lors que la profondeur vaut 2.

Apprentissage d'un arbre de régression

Détaillons la procédure d'apprentissage d'un arbre de régression. On suppose que l'on considère une seule variable pour nos observations X. On cherche à trouver la coupure t_1 optimale.

Notons $R_G = \{X : X \leq t_1\}$ l'ensemble à gauche de t_1 et $R_D = \{X : X > t_1\}$ l'ensemble à droite de t_1. Minimiser la perte quadratique revient à résoudre

$$\min_{t_1 \in \mathbb{R}} \sum_{i=1}^{n} (y_i - \hat{y})^2$$

Apprentissage d'un arbre de régression

Seulement, t_1 n'apparaît pas dans la fonction de perte. On joue alors sur les indices en considérant d'une part les points dans R_G, et d'autre part les points dans R_D.

De plus, puisque l'on souhaite attribuer un poids à chaque noeud, qui ici définira la valeur prédite $\hat{y}$, nous devons introduire c_G et c_D, que l'on interprète comme ceci :

- Tous les points X qui appartiennent à R_G (soit inférieurs à t_1) auront pour valeur prédite c_G.
- Tous les points X qui appartiennent à R_D (soit strictement supérieurs à t_1) auront pour valeur prédite c_D.

Apprentissage d'un arbre de régression

Mais ces poids doivent aussi être optimisés afin d'être calibré selon nos données. Dès lors, on obtient une formule beaucoup plus applicable numériquement :

$$\min_{t_1 \in \mathbb{R}} \left(\min_{c_G \in \mathbb{R}} \frac{1}{|R_G|} \sum_{x_i \in R_G} (y_i - c_G)^2 + \min_{c_D \in \mathbb{R}} \frac{1}{|R_D|} \sum_{x_i \in R_D} (y_i - c_D)^2 \right)$$

Par chance, on peut montrer que pour une fonction de perte quadratique, c_G et c_D peuvent s'estimer directement :

$$\hat{c_G} = \mathbb{E}[Y|X \in R_G] \qquad \hat{c_D} = \mathbb{E}[Y|X \in R_D]$$

Et donc le problème se simplifie en :

$$\min_{t_1 \in \mathbb{R}} \left(\frac{1}{|R_G|} \sum_{x_i \in R_G} (y_i - \hat{c_G})^2 + \frac{1}{|R_D|} \sum_{x_i \in R_D} (y_i - \hat{c_D})^2 \right)$$

Apprentissage d'un arbre de régression

En résumé, pour chaque phase de coupure, il faut :

- Trouver sur quelle variable effectuer la coupure
- Fixer un point de coupure t
- Calculer les moyennes des deux régions R_G et R_D

$$\min_{1 \leq j \leq p} \min_{t \in \mathbb{R}} \left(\frac{1}{|R_G|} \sum_{x_i \in R_G} (y_i - \hat{c_G})^2 + \frac{1}{|R_D|} \sum_{x_i \in R_D} (y_i - \hat{c_D})^2 \right)$$

Un algorithme *greedy* possède une complexité dans le pire des cas à $O(pn^2)$. Les implémentations actuelles permettent d'obtenir une complexité $O(pn \log n)$ par tri ascendant.

Apprentissage d'un arbre de classification

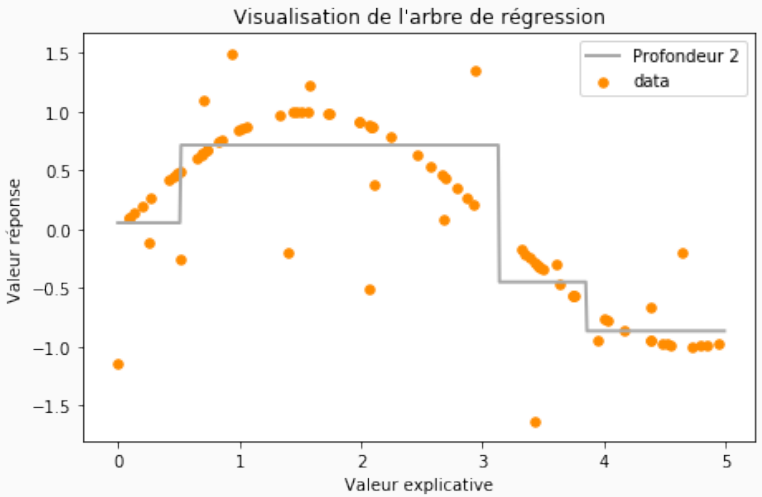
Pour de la classification, le principe est le même. Seule la procédure d'optimisation de la coupure change afin de prendre en compte le caractère discret des labels. On utilise régulièrement l'indice de Gini comme critère de minimisation de la perte (aussi appelé risque).

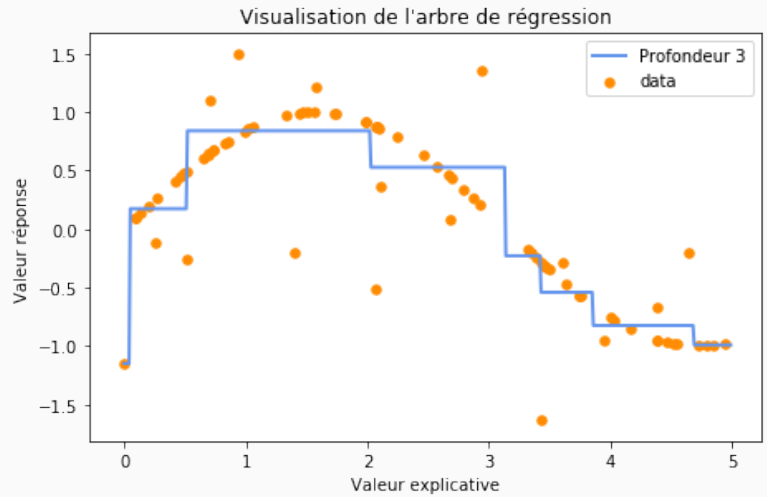
$$\min_{1\leq j\leq p}\min_{t\in\mathbb{R}}\left(\sum_{k=1}^{K}\hat{p}_{Gk}(1-\hat{p}_{Gk})+\sum_{k=1}^{K}\hat{p}_{Dk}(1-\hat{p}_{Dk})\right)$$

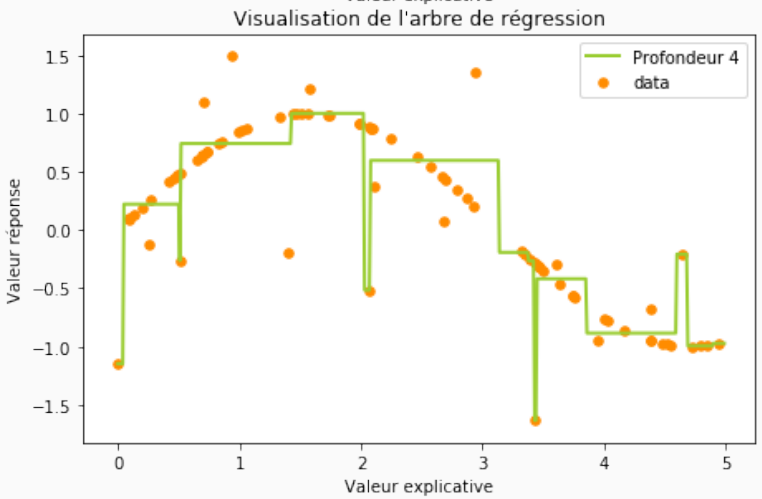
Avec

$$\hat{p}_{Gk}=\frac{1}{|R_G|}\sum_{x_i\in R_G}\mathbf{1}_{\{y_i=k\}}\text{ et }\hat{p}_{Dk}=\frac{1}{|R_D|}\sum_{x_i\in R_D}\mathbf{1}_{\{y_i=k\}}$$

Sur-apprentissage







TP

TP Jupyter : CART