Programmation d'une Intelligence Artificielle : arbres ou apprentissage automatique

Arbres de Décision.

Paul Mangold

L3 MIASHS

19 Mars 2021

Machine Learning

Imaginons que l'on a quelques informations :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

$\rightarrow$ Comment deviner quel fruit on a dans la main avec couleur/diamètre/noyau ?

Machine Learning

Des fruits...

On connaît : une série d'**exemples**.

On cherche : à reconnaître de **futurs exemples**.

$\rightarrow$ Machine Learning Supervisé !

Revenons à nos fruits :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

On peut remarquer que :

- si le diamètre est 2, c'est une cerise ;

Revenons à nos fruits :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

On peut remarquer que :

- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;

Revenons à nos fruits :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

On peut remarquer que :

- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;
- si le diamètre est 5, c'est une pomme ou un citron ;

Revenons à nos fruits :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

On peut remarquer que :

- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;
- si le diamètre est 5, c'est une pomme ou un citron ;
- etc.

Disons que je vous donne un nouveau fruit inconnu :

Bon ok vous savez ce que c'est... mais jouons le jeu, on connaît :

Couleur	Diamètre	Noyau	Fruit
Jaune	3	Non	?

Il nous manque la réponse :

Couleur	Diamètre	Noyau	Fruit
Jaune	3	Non	?

Il nous manque la réponse :

Couleur	Diamètre	Noyau	Fruit
Jaune	3	Non	?

Reprenons les règles que l'on avait apprises :

- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;
- si le diamètre est 5, c'est une pomme ou un citron.

$\rightarrow$ c'est une banane !

Ok trop facile, en voilà un autre :

On connaît :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	?

On connaît :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	?

Reprenons les règles que l'on avait apprises :

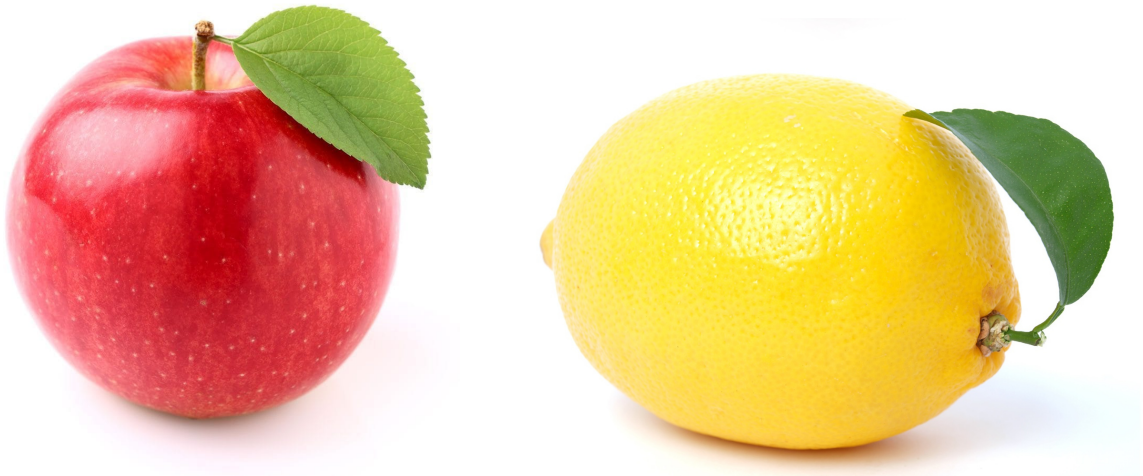
- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;
- si le diamètre est 5, c'est une pomme ou un citron.

$\rightarrow$ c'est une pomme... ou un citron.

Premier problème :

nos règles ne sont pas assez "puissantes".

→ on ne différencie pas les pommes des citrons.



Bon pourtant c'est pas la même chose.

On va rajouter une règle :

- si le diamètre est 2, c'est une cerise ;
- si le diamètre est 3, c'est une banane ;
- si le diamètre est 5 alors :
  - si c'est jaune, c'est un citron ;
  - si c'est vert c'est une pomme.
  - si c'est rouge c'est une pomme.

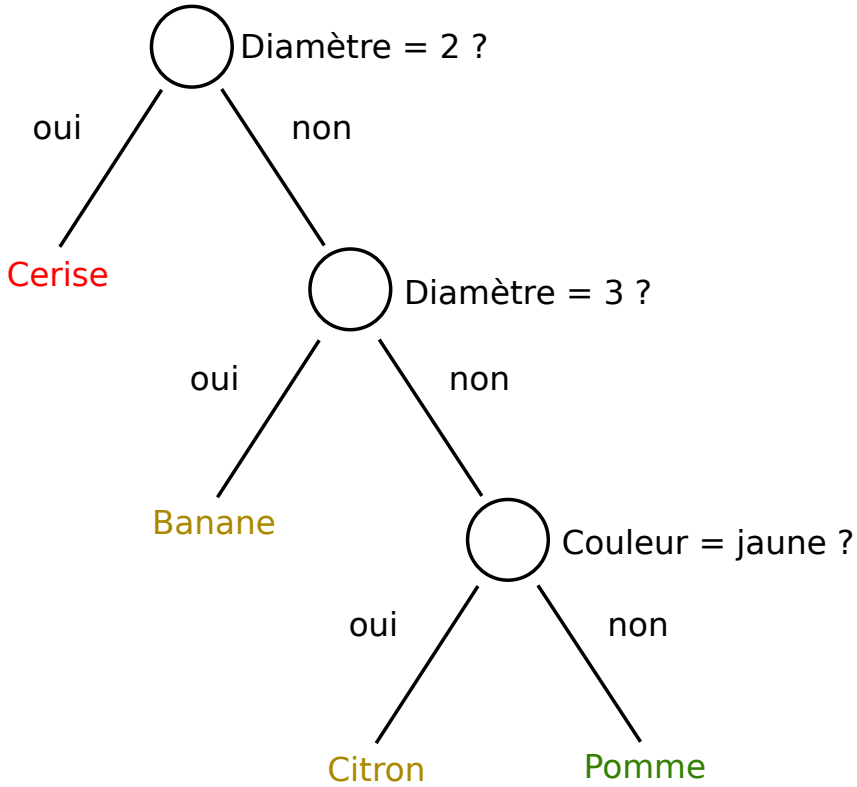
Rééssayons :

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	?

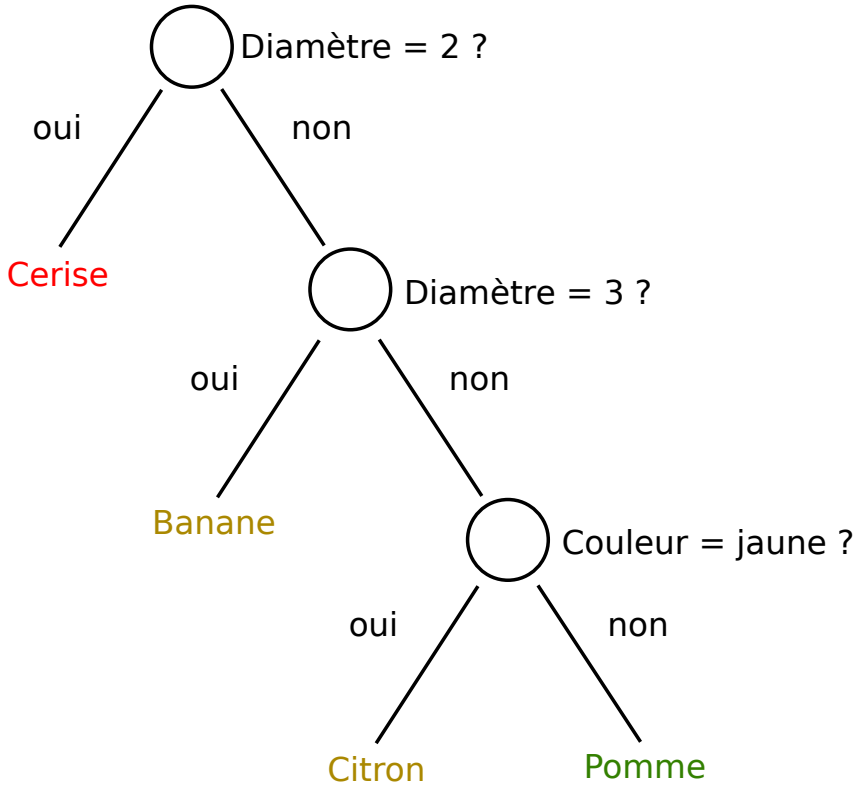
Cette fois on trouve la bonne réponse.

On peut représenter nos règles sous la forme d'un arbre où :

- chaque *sommet* est une question ;
- chaque *feuille* est un label.



C'est un **arbre de décision**.



→ Mais comment apprendre un tel arbre ?

Définissons mieux notre problème...

Couleur	Diamètre	Noyau	Fruit
Rouge	5	Non	Pomme
Vert	5	Non	Pomme
Jaune	3	Non	Banane
Jaune	5	Non	Citron
Rouge	2	Oui	Cerise

Dans notre base de donnée on avait :

- des **features** : couleur, diamètre, noyaux ;
- des **labels** : fruit.

On suppose donc qu'on a deux ensembles :

- un ensemble $\mathcal{X}$ de features ;
- un ensemble $\mathcal{Y}$ de labels.

On cherche à apprendre une fonction $h : \mathcal{X} \mapsto \mathcal{Y}$ qui

- prend un ensemble de features en entrée ;
- renvoie un label.

Par exemple on avait avec nos règles :

$$h\left(\begin{array}{c|c|c|c} \text{Couleur} & \text{Diamètre} & \text{Noyau} & \text{Fruit} \\ \hline \text{Jaune} & 5 & \text{Non} & ? \end{array}\right) = \textit{Citron}.$$

La fonction h **est déterminée par notre arbre de décision**.

$\rightarrow$ on a donc défini une fonction qui permet de classifier des points.

On peut regarder les prédictions de h sur notre base de données :

Couleur	Diamètre	Noyau	Fruit	**h**
Rouge	5	Non	Pomme	**Pomme**
Vert	5	Non	Pomme	**Pomme**
Jaune	3	Non	Banane	**Banane**
Jaune	5	Non	Citron	**Citron**
Rouge	2	Oui	Cerise	**Cerise**

Donc notre fonction reconnaît correctement nos fruits.

On peut mesurer la précision de notre fonction :

$$precision(h) = \frac{\text{nombre de prédictions correctes}}{\text{nombre de prédictions}}.$$

Dans notre cas on a 5 prédictions, dont 5 prédictions correctes.

$$h\left(\begin{array}{c|c|c} \text{Couleur} & \text{Diamètre} & \text{Noyau} \\ \hline \text{Jaune} & 5 & \text{Non} \\ \text{Vert} & 5 & \text{Non} \\ \text{Jaune} & 3 & \text{Non} \\ \text{Jaune} & 5 & \text{Non} \\ \text{Rouge} & 2 & \text{Oui} \end{array}\right) = \begin{array}{c} \text{Fruit} \\ \hline \text{Pomme} \\ \text{Pomme} \\ \text{Banane} \\ \text{Citron} \\ \text{Cerise} \end{array} \quad (1)$$

Donc $precision(h) = 1$.

Mais si on avait

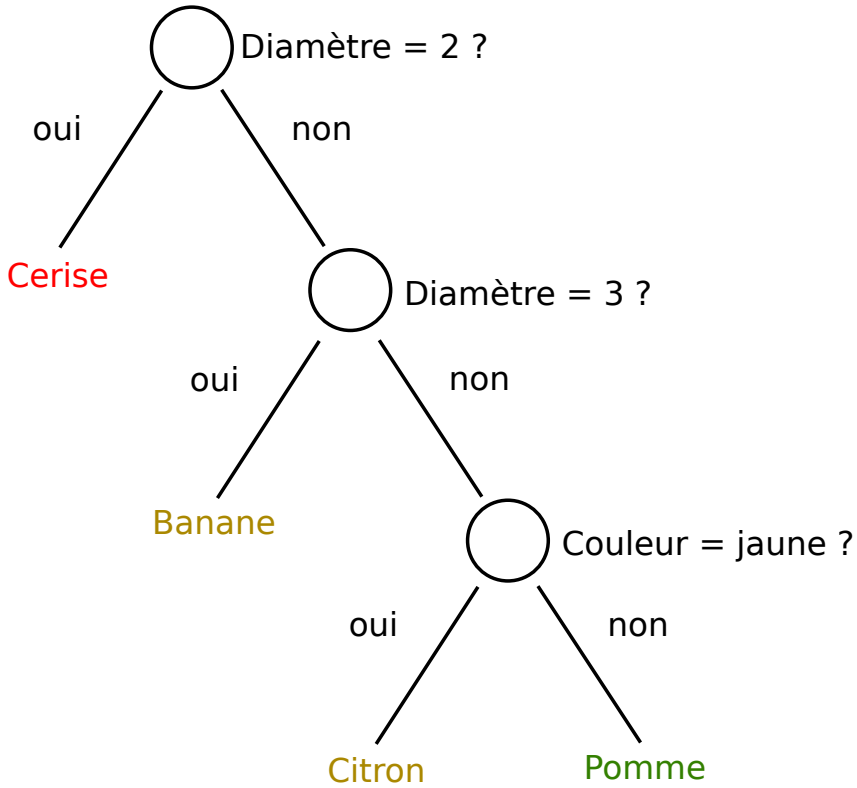
$$h\left(\begin{array}{c|c|c} \text{Couleur} & \text{Diamètre} & \text{Noyau} \\ \hline \text{Jaune} & 5 & \text{Non} \\ \text{Vert} & 5 & \text{Non} \\ \text{Jaune} & 3 & \text{Non} \\ \text{Jaune} & 5 & \text{Non} \\ \text{Rouge} & 2 & \text{Oui} \end{array}\right) = \begin{array}{c} \text{Fruit} \\ \hline \text{Citron} \\ \text{Pomme} \\ \text{Banane} \\ \text{Citron} \\ \text{Cerise} \end{array} \quad (2)$$

Alors $\mathit{precision}(h) = \frac{4}{5} = 0.8$.

Question : comment choisir nos questions pour avoir une précision maximale ?

Apprentissage d'Arbres de Décision

À chaque sommet on pose une question :



Des Questions ?

On avait une base de donnée avec des points $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

Chaque élément de x est soit :

- un nombre (par ex, $\mathit{diametre} = 5$ ou $\mathit{diametre} \leq 5$) ;
- une catégorie (par ex, $\mathit{couleur} = \mathit{jaune}$).

Pour un élément x_i de x, on cherche des questions de la forme :

- si x_i est un nombre, pour un certain $a \in \mathbb{R}$:

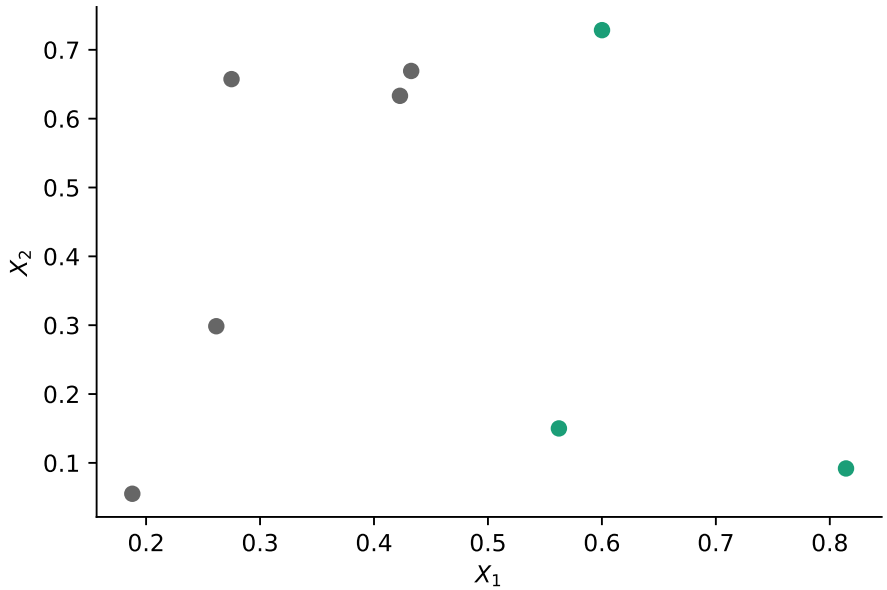
$$x_i \leq a;$$

- si x_i est une catégorie, pour un ensemble de catégories C :

$$x_i \in C.$$

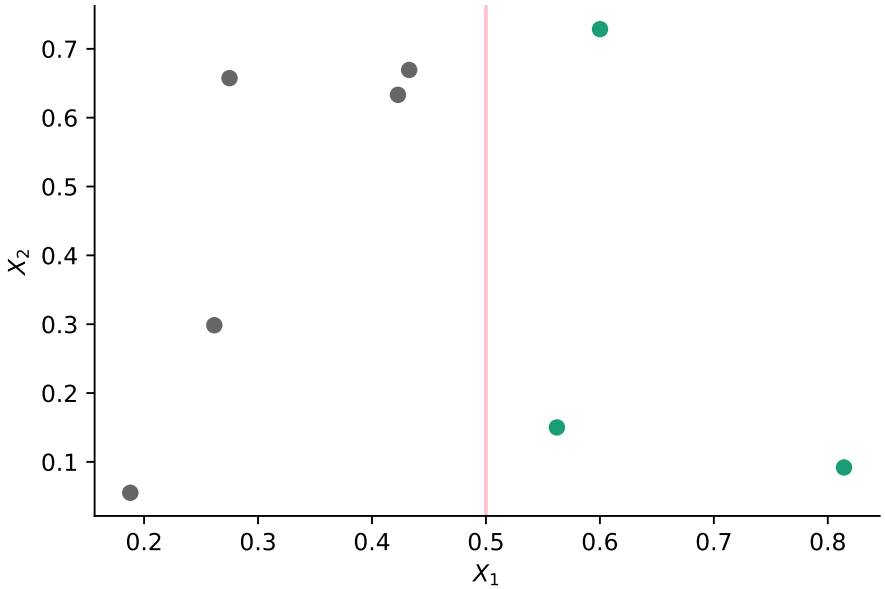
Chaque question permet de diviser l'espace en deux !

Par exemple, supposons que l'on a les points suivants :



On peut poser la question

$$\text{“}X_1 < 0.5 \text{ ?”}$$



qui divise correctement notre espace !

Mais comment trouver des bonnes questions ?

$\rightarrow$ on doit pouvoir les **évaluer**.

Une méthode est de calculer le score Gini.

Le score Gini mesure l'homogénéité (ou la "pureté") d'un ensemble.

Par exemple, l'ensemble

$$y = [0, 0, 0, 0, 0, 1]$$

est plutôt "pur" car ses éléments ont presque tous la même valeur.

Mais l'ensemble

$$y = [0, 5, 3, 2, 8, 1]$$

est très "impur" car ses éléments ont tous des valeurs différentes.

Prenons notre ensemble

$$y = [Pomme, Pomme, Banane, Citron, Cerise]$$

Pour chaque valeur on calcule :

$$P_{Pomme} = \frac{\text{nombre d'occurences de "Pomme" dans } y}{\text{nombre d'éléments dans } y}$$

Cela correspond à la probabilité de classifier notre pomme correctement si :

- on prend une pomme dans notre ensemble ;
- on choisit une valeur au hasard de l'ensemble ;
- on regarder si cette valeur est "Pomme" ou non.

L'intuition derrière cette définition est que :

- si on n'a que des Pommes, on ne se trompe jamais ;
- sinon, moins on a de Pommes, plus on se trompe.

Le score Gini est alors défini ainsi :

$$Gini(y) = \sum_{c \in C} P_c(1 - P_c), \tag{3}$$

où C est l'ensemble des valeurs uniques apparaissant dans y.

On va manipuler ce score en TP.

Revenons à nos arbres : on a des features X et des labels y.

On pose une question, par exemple "$X_1 < 0.5$?", cela divise nos ensembles en deux :

- d'un côté, les points où $X_1 < 0.5$;
- de l'autre, les points où $X_1 \geq 0.5$.

Revenons à nos arbres : on a des features X et des labels y.

On pose une question, par exemple "$X_1 < 0.5$?", cela divise nos ensembles en deux :

- d'un côté, les points où $X_1 < 0.5$;
- de l'autre, les points où $X_1 \geq 0.5$.

On a donc divisé les labels en deux ensembles $y_{<0.5}$ et $y_{\geq 0.5}$.

Revenons à nos arbres : on a des features X et des labels y.

On pose une question, par exemple "$X_1 < 0.5$?", cela divise nos ensembles en deux :

- d'un côté, les points où $X_1 < 0.5$;
- de l'autre, les points où $X_1 \geq 0.5$.

On a donc divisé les labels en deux ensembles $y_{<0.5}$ et $y_{\geq 0.5}$.

$\rightarrow$ On peut calculer les scores Gini de $y_{<0.5}$ et $y_{\geq 0.5}$!

On peut ainsi définir le score d'une question :

- au début, on a un ensemble de labels y à séparer dont le score est

$$Gini(y);$$

- après notre question, on a deux ensembles (ici $y_{<0.5}$ et $y_{\geq 0.5}$). Le score est alors défini comme :

$$Gini(question) = \frac{|y_{<0.5}|}{|y|} Gini(y_{<0.5}) + \frac{|y_{\geq 0.5}|}{|y|} Gini(y_{\geq 0.5}),$$

c'est la moyenne pondérée des deux scores obtenus !

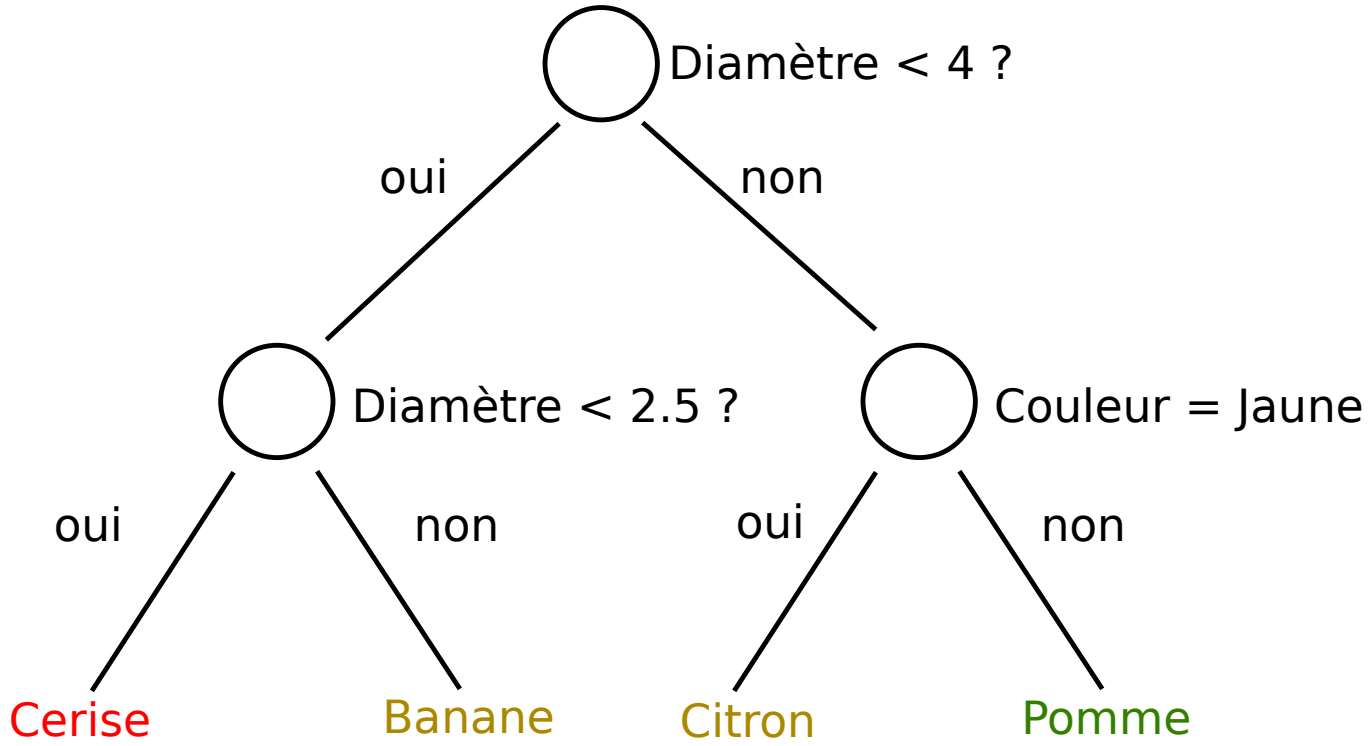
On peut donc donner un score à chaque question...

$\rightarrow$ puis on choisit la question qui fait le plus baisser le score.
$\rightarrow$ et ainsi de suite jusqu'à ce que soit :

- nos questions permettent de classifier tous les exemples que l'on connait ;
- on n'a plus de questions à poser.

C'est l'algortihme CART.

Par exemple dans le cas ci-dessus :



TP !

TP !

Dans le TP d'aujourd'hui on va :

- comprendre le score Gini ;
- apprendre des arbres de décision ;
- voir les limites de ces arbres et comment les dépasser.