# 1. Apprentissage

## 1.1. Algorithme

On considère que l'algorithme peut être définie par la définition suivante :

**Définition 1.1.** Prédiction a partir de différents questions aux attributs des données en les divisant à chaque décision. On considère la nomenclature :

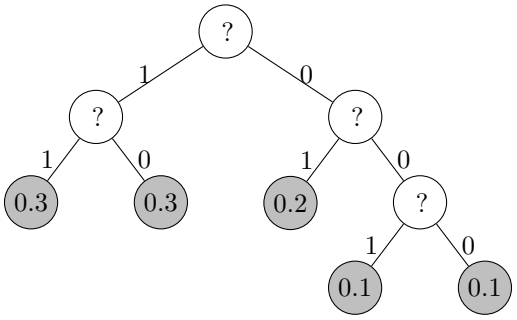


FIGURE 1.1 : Representation Arbre de Décision

On considère :

1. **node** :
   (a) `root` : première node, représente tous les données ;
   (b) `decision` : sub-node qui se divide en d'autre sub-nodes en représentant une question ;
   (c) `leaf` : node qui ne se divide pas, présente une partie des données ;
2. **splitting** : process division d'une node en sub-nodes ;
3. **pruning** : process d'enlever une node ;

On considère pour convention :

1. **gauche** représente réponse `true`, ou 1 ;
2. **droite** représente réponse `false`, ou 0 ;

On considère que la profondeur p d'une arbre de décision est simply the largest possible length between the root to a leaf.

**Remarque.** maximum depth would be N-1, where N is the number of training samples. You can derive this by considering that the least effective split would be peeling off one training example per node.

**Remarque.** L'Arbre de Décision n'est pas trop sensible aux données mais de toute façon ont tendance d'avoir une variance élevée.

On classifiée l'Arbre de Décision en :

1. **Classification** : les outputs sont des categories ;
2. **Regression** : les outputs sont des numéros ;

À chaque noeud on choisit le test $T$ qui maximise le gain en homogénéité :

$$T = \arg_T \max(Gain(D, T)) \tag{1.1}$$

Où :

1. $D$ : population de données ;
2. $Gain(D,T)$ : gain of the test T in the population D :

$$Gain(D,T) = I(D) - \sum_i (p(D_i|D,T) \cdot I(D_i)) \qquad \text{où} \qquad p(D_i|D,T) = \frac{|D_i|}{|D|} \tag{1.2}$$

Où :

(a) $I(D)$ : Critère d'Hétérogénéité ;
(b) $p(D_i|D,T)$ : Proportion de Données dans D sélectionnées par la branche $i$ ;

La décision sera fait à partir d'une critère arbitraire, entre les plus communs on a :

1. **Indice de Gini** : seulement pour une arbre binaire ;

$$I(D) = \sum_i p_i(D) \cdot (1 - p_i(D)) \tag{1.3}$$

2. **Indice d'Erreur** ;

$$I(D) = 1 - \max_i(p_i(D)) \tag{1.4}$$

3. **Entropie** ;

$$I(D) = -\sum_i p_i(D) \cdot \log_2(p_i(D)) \tag{1.5}$$

Parfois sera difficile de trouver les valeurs pour chaque test car il faut beaucoup d'operations pour y arriver. On peut considérer que **maximiser** le $Gain(D,T)$ c'est suffisant de **minimiser** le Critère d'Hétérogénéité.

À la fin on aura une division des données qui peut se rassembler à le suivant :

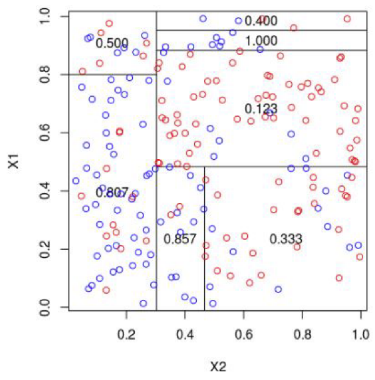


FIGURE 1.2 : Representation Division

La choix de la profondeur p de l'Arbre de Décision déterminera la qualité de la prédiction. Si on augment p la prediction améliore. Généralement on aura le comportement suivant :

| | | |
|---|---|---|
| p ↑ | biais ↓ | variance ↑ |
| p ↓ | biais ↑ | variance ↓ |

TABLE 1.1 : Comportement Decision Tree

Comme les Arbres de Décision ont une tendance au surapprentissage on peut utiliser des **Forêts Aléatoires** :

**Définition 1.2.** Construction de plusieurs arbres se basant non seulement sur des échantillons différents mais aussi sur des variables différents. On considère le diagramme :

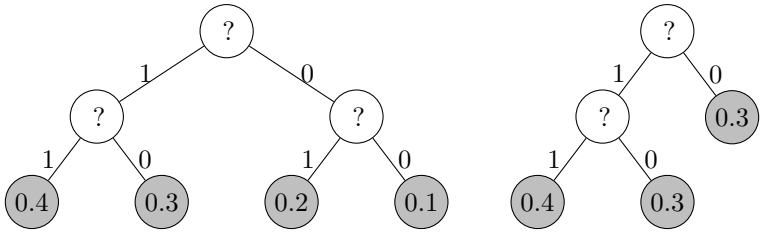


FIGURE 1.3 : Representation Forêt de Décision

La décision finale sera fait avec le vote majoritaire.

### 1.1.1. Avantages

On peut citer :

1. representation facile à comprendre ;

### 1.1.2. Incovenients

On peut citer :

1. tendance au sur-apprentissage ;

### 1.1.3. Applications

Cet algorithme n'est que utilise dans le cadre de problèmes de classification.