LA BRIQUE ÉLÉMENTAIRE: L'ARBRE DE DÉCISION

Les arbres de décision sont des outils puissants en apprentissage automatique, utilisés pour des tâches de classification et de régression. Ces algorithmes non paramétriques consistent à diviser l'espace des caractéristiques en sous-ensembles homogènes à l'aide de règles simples, afin de faire des prédictions. Malgré leur simplicité apparente, les arbres de décision sont capable de saisir des relations complexes et non linéaires entre les variables (ou *caractéristiques*) d'un jeu de données.

1. Le principe fondamental : partitionner pour prédire

Imaginez que vous souhaitiez prédire le prix d'une maison en fonction de sa superficie et de son nombre de pièces. L'espace des caractéristiques (superficie et nombre de pièces) est vaste, et les prix des maisons (la *réponse* à prédire) sont très variables. Pour prédire le prix des maisons, l'idée est de diviser cet espace en zones plus petites, où les maisons ont des prix similaires, et d'attribuer une prédiction identique à toutes les maisons situées dans la même zone.

1.1. Les défis du partitionnement optimal.

L'objectif principal est de trouver la partition de l'espace des caractéristiques qui offre les meilleures prédictions possibles. Cependant, cet objectif se heurte à plusieurs difficultés, et la complexité du problème augmente rapidement avec le nombre de caractéristiques et la taille de l'échantillon:

1. **Infinité des découpages possibles** : Il existe une infinité de façons de diviser l'espace des caractéristiques.
2. **Complexité de la paramétrisation** : Il est difficile de représenter tous ces découpages avec un nombre limité de paramètres.
3. **Optimisation complexe** : Même avec une paramétrisation, trouver le meilleur découpage nécessite une optimisation complexe, souvent irréaliste en pratique.

1.2. Les solutions apportées par les arbres de décision.

Pour surmonter ces difficultés, les méthodes d'arbres de décision, et notamment la plus célèbre, l'algorithme CART (Classication And Regression Tree, Breiman et al. (1984)), adoptent deux approches clés :

1. **Simplification du partitionnement de l'espace**

Au lieu d'explorer tous les découpages possibles, les arbres de décision partitionnent l'espace des caractéristiques en plusieurs régions distinctes (non chevauchantes) en appliquant des règles de décision simples. Les règles suivantes sont communément adoptées:

- **Découpages binaires simples** : À chaque étape, l'algorithme divise une région de l'espace en deux sous-régions en se basant sur une seule caractéristique (ou *variable*) et en définissant un seul seuil (ou *critère*) pour cette segmentation. Concrètement, cela revient à poser une question du type : "La valeur de la caractéristique X dépasse-t-elle un certain seuil ?" Par exemple : "La superficie de la maison est-elle supérieure à 100 m^2 ?". Les deux réponses possibles ("Oui" ou "Non") génèrent deux nouvelles sous-régions distinctes de l'espace, chacune correspondant à un sous-ensemble de données plus homogène.
- **Prédictions locales** : Lorsque l'algorithme s'arrête, une prédiction simple est faite dans chaque région. Il s'agit souvent de la moyenne des valeurs cibles dans cette région (régression) ou de la classe majoritaire (classification).

Ces règles de découpage rendent le problème d'optimisation plus simple mais également plus interprétable.

2. **Optimisation gloutonne (greedy)**

Plutôt que d'optimiser toutes les divisions simultanément, les arbres de décision utilisent une approche simplifiée, récursive et séquentielle :

- **Division étape par étape** : À chaque étape, l'arbre choisit la meilleure division possible sur la base d'un critère de réduction de l'hétérogénéité intra-région. En revanche, il ne prend pas en compte les étapes d'optimisation futures.
- **Critère local** : La décision est basée sur la réduction immédiate de l'impureté ou de l'erreur de prédiction (par exemple, la réduction de la variance pour la régression). Ce processus est répété pour chaque sous-région, ce qui permet d'affiner progressivement la partition de l'espace en fonction des caractéristiques les plus discriminantes.

Cette méthode dite "gloutonne" (*greedy*) s'avère efficace pour construire un partitionnement de l'espace des caractéristiques, car elle décompose un problème d'optimisation complexe en une succession de problèmes plus simples et plus rapides à résoudre. Le résultat obtenu n'est pas nécessairement un optimum global, mais il s'en approche raisonnablement et surtout rapidement.

Le terme "arbre de décision" provient de la structure descendante en forme d'arbre inversé qui émerge lorsqu'on utilise un algorithme glouton pour découper l'espace des caractéristiques en sous-ensemble de réponses homogènes de manière récursive. A chaque étape, deux nouvelles branches sont créées et forment une nouvelle partition de l'espace des caractéristiques.

Une fois entraîné, un arbre de décision est une fonction **constante par morceaux** défini sur l'espace des caractéristiques. En raison de leur nature **non-continue** et **non-différentiable**, il est impossible d'utiliser des méthodes d'optimisation classiques reposant sur le calcul de gradients.

1.3. **Terminologie et structure d'un arbre de décision.**

Nous présentons la structure d'un arbre de décision et les principaux éléments qui le composent.

- **Nœud Racine (Root Node)** : Le nœud racine est le point de départ de l'arbre de décision, il est situé au sommet de l'arbre. Il contient l'ensemble des données d'entraînement avant toute division. À ce niveau, l'algorithme cherche la caractéristique la plus discriminante, c'est-à-dire celle qui permet de diviser les données de manière à optimiser une fonction de perte (comme l'indice de Gini pour la classification ou la variance pour la régression).
- **Nœuds Internes (Internal Nodes)** : Les nœuds internes sont les points intermédiaires où l'algorithme CART applique des règles de décision pour diviser les données en sous-ensembles plus petits. Chaque nœud interne représente une question ou condition basée sur une caractéristique particulière (par exemple, "La superficie de la maison est-elle supérieure à 100 m^2 ?"). À chaque étape, une seule caractéristique (la superficie) et un seul seuil (supérieur à 100) sont utilisés pour faire la division.
- **Branches**: Les branches sont les connexions entre les nœuds, elles illustrent le chemin que les données suivent en fonction des réponses aux questions posées dans les nœuds internes. Chaque branche correspond à une décision binaire, "Oui" ou "Non", qui oriente les observations vers une nouvelle subdivision de l'espace des caractéristiques.
- **Nœuds Terminaux ou Feuilles (Leaf Nodes ou Terminal Nodes)** : Les nœuds terminaux, situés à l'extrémité des branches, sont les points où le processus de division s'arrête. Ils fournissent la prédiction finale.

- En **classification**, chaque feuille correspond à une classe prédite (par exemple, “Oui” ou “Non”).
- En **régression**, chaque feuille fournit une valeur numérique prédite (comme le prix estimé d’une maison).

Figure illustrative : Une représentation visuelle de la structure de l’arbre peut être utile ici pour illustrer les concepts de nœuds, branches et feuilles.

1.4. **Illustration.**

Supposons que nous souhaitions prédire le prix d’une maison en fonction de sa superficie et de son nombre de pièces. Un arbre de décision pourrait procéder ainsi :

1. **Première division** : “La superficie de la maison est-elle supérieure à 100 m² ?”
   - Oui : Aller à la branche de gauche.
   - Non : Aller à la branche de droite.
2. **Deuxième division (branche de gauche)** : “Le nombre de pièces est-il supérieur à 4 ?”
   - Oui : Prix élevé (par exemple, plus de 300 000 €).
   - Non : Prix moyen (par exemple, entre 200 000 € et 300 000 €).
3. **Deuxième division (branche de droite)** : “Le nombre de pièces est-il supérieur à 2 ?”
   - Oui : Prix moyen (par exemple, entre 150 000 € et 200 000 €).
   - Non : Prix bas (par exemple, moins de 150 000 €).

Cet arbre utilise des règles simples pour diviser l’espace des caractéristiques (superficie et nombre de pièces) en sous-groupes homogènes et fournir une prédiction (estimer le prix d’une maison).

Figure illustrative

2. L’algorithme CART, un partitionnement binaire récursif

L’algorithme CART (Classification and Regression Trees) proposé par Breiman, Friedman, Olshen, & Stone (1984) est une méthode utilisée pour construire des arbres de décision, que ce soit pour des tâches de classification ou de régression. L’algorithme CART fonctionne en partitionnant l’espace des caractéristiques en sous-ensembles de manière récursive, en suivant une logique de décisions binaires à chaque étape. Ce processus est itératif et suit plusieurs étapes clés.

2.1. **Définir une fonction d'impureté adaptée au problème.**

La **fonction d'impureté** est une mesure locale utilisée dans la construction des arbres de décision pour évaluer la qualité des divisions à chaque nœud. Elle quantifie le degré d'hétérogénéité des observations dans un nœud par rapport à la variable cible (classe pour la classification, ou valeur continue pour la régression). Plus précisément, une mesure d'impureté est conçue pour croître avec la dispersion dans un nœud. Un nœud est dit **pur** lorsque toutes les observations qu'il contient appartiennent à la même classe (classification) ou présentent des valeurs similaires/identiques (régression).

L'algorithme CART utilise ce type de mesure pour choisir les divisions qui créent des sous-ensembles plus homogènes que le nœud parent. À chaque étape de construction, l'algorithme sélectionne la division qui réduit le plus l'impureté, afin de garantir des nœuds de plus en plus homogènes au fur et à mesure que l'arbre se développe.

Le choix de la fonction d'impureté dépend du type de problème :

- **Classification** : L'**indice de Gini** ou l'**entropie** sont très souvent utilisées pour évaluer la dispersion des classes dans chaque nœud.
- **Régression** : La **somme des erreurs quadratiques** (SSE) est souvent utilisée pour mesurer la variance des valeurs cibles dans chaque nœud.

2.1.1. *Mesures d'impureté classiques pour les problèmes de classification.*

Dans le cadre de la classification, l'objectif est de partitionner les données de manière à ce que chaque sous-ensemble (ou région) soit le plus homogène possible en termes de classe prédite. Plusieurs mesures d'impureté sont couramment utilisées pour évaluer la qualité des divisions.

Propriété-définition d'une mesure d'impureté

Pour un nœud t contenant K classes, une **mesure d'impureté** $I(t)$ est une fonction qui quantifie l'hétérogénéité des classes dans ce nœud. Elle doit satisfaire les propriétés suivantes :

- **Pureté maximale** : Lorsque toutes les observations du nœud appartiennent à une seule classe, c'est-à-dire que la proportion $p_k = 1$ pour une classe k et $p_j = 0$ pour toutes les autres classes $j \neq k$, l'impureté est minimale et $I(t) = 0$. Cela indique que le nœud est **entièrement pur**, ou homogène.

- **Impureté maximale** : Lorsque les observations sont réparties de manière uniforme entre toutes les classes, c'est-à-dire que $p_k = \frac{1}{K}$ pour chaque classe k, l'impureté atteint son maximum. Cette situation reflète une **impureté élevée**, car le nœud est très hétérogène et contient une forte incertitude sur la classe des observations.

1. L'indice de Gini

L'**indice de Gini** est l'une des fonctions de perte les plus couramment utilisées pour la classification. Il mesure la probabilité qu'un individu sélectionné au hasard dans un nœud soit mal classé si on lui attribue une classe au hasard, en fonction de la distribution des classes dans ce nœud.

Pour un nœud t contenant K classes, l'indice de Gini $G(t)$ est donné par :

$$G(t) = 1 - \sum_{k=1}^{K} p_k^2$$

où p_k est la proportion d'observations appartenant à la classe k dans le nœud t.

Critère de choix : L'indice de Gini est souvent utilisé parce qu'il est simple à calculer et capture bien l'homogénéité des classes au sein d'un nœud. Il privilégie les partitions où une classe domine fortement dans chaque sous-ensemble.

2. L'entropie (ou entropie de Shannon)

L'**entropie** est une autre mesure de l'impureté utilisée dans les arbres de décision. Elle mesure la quantité d'incertitude ou de désordre dans un nœud, en s'appuyant sur la théorie de l'information.

Pour un nœud t contenant K classes, l'entropie $E(t)$ est définie par :

$$E(t) = -\sum_{k=1}^{K} p_k \log(p_k)$$

où p_k est la proportion d'observations de la classe k dans le nœud t.

Critère de choix : L'entropie a tendance à être plus sensible aux changements dans les distributions des classes que l'indice de Gini, car elle attribut un poids plus élevé aux événements rares (valeurs de p_k très faibles). Elle est souvent utilisée lorsque l'erreur de classification des classes minoritaires est particulièrement importante.

3. Taux d'erreur

Le **taux d'erreur** est une autre mesure de l'impureté parfois utilisée dans les arbres de décision. Il représente la proportion d'observations mal classées dans un nœud.

Pour un nœud t, le taux d'erreur TE (t) est donné par :

$$\text{TE } (t) = 1 - \max(p_k)$$

où $\max(p_k)$ est la proportion d'observations appartenant à la classe majoritaire dans le nœud.

Critère de choix : Bien que le taux d'erreur soit simple à comprendre, il est moins souvent utilisé dans la construction des arbres de décision parce qu'il est moins sensible que l'indice de Gini ou l'entropie aux petits changements dans la distribution des classes.

2.1.2. *Mesures d'impureté classiques pour les problèmes de régression.*

Dans les problèmes de régression, l'objectif est de partitionner les données de manière à réduire au maximum la variabilité des valeurs au sein de chaque sous-ensemble. Pour mesurer cette variabilité, la somme des erreurs quadratiques (SSE) est la fonction d'impureté la plus couramment employée. Elle évalue l'impureté d'une région en quantifiant à quel point les valeurs de cette région s'écartent de la moyenne locale.

1.Somme des erreurs quadratiques (SSE) ou variance

La **somme des erreurs quadratiques** (ou **SSE**, pour *Sum of Squared Errors*) est une mesure qui quantifie la dispersion des valeurs dans un nœud par rapport à la moyenne des valeurs dans ce nœud.

Formule : Pour un nœud t, contenant N observations avec des valeurs y_i, la SSE est donnée par :

$$\text{SSE } (t) = \sum_{i=1}^{N} (y_i - \hat{y})^2$$

où $\hat{y}$ est la moyenne des valeurs y_i dans le nœud t.

Propriété :

- Si toutes les valeurs de y_i dans un nœud sont proches de la moyenne $\hat{y}$, la SSE sera faible, indiquant une homogénéité élevée dans le nœud.
- En revanche, une SSE élevée indique une grande variabilité dans les valeurs, donc un nœud impur.

Critère de choix : La somme des erreurs quadratiques (SSE) est particulièrement sensible aux écarts élevés entre les valeurs observées et la moyenne prédite. En cherchant à minimiser la SSE, les modèles visent à former des nœuds dans lesquels les valeurs des observations sont aussi proches que possible de la moyenne locale.

2.2. **Identifier la partition binaire maximisant la réduction de l'impureté.**

Une fois la mesure d'impureté définie, l'algorithme CART examine toutes les divisions binaires possibles de l'espace des caractéristiques. À chaque nœud, et pour chaque caractéristique, il cherche à identifier le **seuil optimal**, c'est-à-dire le seuil qui minimise le plus efficacement l'impureté des deux sous-ensembles générés. L'algorithme compare ensuite toutes les divisions potentielles (caractéristiques et seuils optimaux associés à chaque nœud) et sélectionne celle qui entraîne la réduction maximale de l'impureté.

Prenons l'exemple d'une caractéristique continue, telle que la superficie d'une maison :

- Si l'algorithme teste la règle "Superficie > 100 m^2", il calcule la fonction de perte pour les deux sous-ensembles générés par cette règle ("Oui" et "Non").
- Ce processus est répété pour différentes valeurs seuils afin de trouver la partition qui minimise le plus efficacement l'impureté au sein des sous-ensembles.

2.3. **Réitérer le processus jusqu'à atteindre un critère d'arrêt.**

L'algorithme CART poursuit le partitionnement de l'espace des caractéristiques en appliquant de manière récursive les mêmes étapes : identification de la caractéristique et du seuil optimal pour chaque nœud, puis sélection du partitionnement binaire qui maximise la réduction de l'impureté. Ce processus est répété jusqu'à ce qu'un **critère d'arrêt** soit atteint, par exemple :

- **Profondeur maximale de l'arbre** : Limiter le nombre de divisions successives pour éviter un arbre trop complexe.
- **Nombre minimum d'observations par feuille** : Empêcher la création de feuilles contenant très peu d'observations, ce qui réduirait la capacité du modèle à généraliser.
- **Réduction minimale de l'impureté à chaque étape**

2.4. **Elagage (*pruning*).**

2.5. **Prédire.**

Une fois l'arbre construit, la prédiction pour une nouvelle observation s'effectue en suivant les branches de l'arbre, en partant du nœud racine jusqu'à un nœud terminal

(ou feuille). À chaque nœud interne, une décision est prise en fonction des valeurs des caractéristiques de l'observation, ce qui détermine la direction à suivre vers l'un des sous-ensembles. Ce cheminement se poursuit jusqu'à ce que l'observation atteigne une feuille, où la prédiction finale est effectuée.

- En **classification**, la classe attribuée est celle majoritaire dans la feuille atteinte.
- En **régression**, la valeur prédite est généralement la moyenne des valeurs cibles des observations dans la feuille.

2.6. **Critères de qualité et ajustements.**

Pour améliorer la performance de l'arbre, on peut ajuster les hyperparamètres tels que la profondeur maximale ou le nombre minimum d'observations dans une feuille. De plus, des techniques comme la **prédiction avec arbres multiples** (bagging, forêts aléatoires) permettent de surmonter les limites des arbres individuels, souvent sujets au surapprentissage.

3. Avantages et limites de cette approche

3.1. **Avantages.**

- **Interprétabilité** : Les arbres de décision sont faciles à comprendre et à visualiser.
- **Simplicité** : Pas besoin de transformations complexes des données.
- **Flexibilité** : Ils peuvent gérer des caractéristiques numériques et catégorielles, ainsi que les valeurs manquantes.
- **Gestion des interactions** : Modèles non paramétriques, pas d'hypothèses sur les lois par les variables. Ils capturent naturellement les interactions entre les caractéristiques.

3.2. **Limites.**

- **Surapprentissage** : Les arbres trop profonds peuvent surapprendre les données d'entraînement.
- **Optimisation locale** : L'approche gloutonne peut conduire à des solutions sous-optimales globalement (optimum local).

- **Stabilité** : De petits changements dans les données peuvent entraîner des changements significatifs dans la structure de l'arbre (manque de robustesse).

References

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Cart. *Classification and Regression Trees*.