Arbre de classification :

* Expliquer Y qualitative, en fct des x_j quelconque, à K modalités

Principe :

A chaque noeud terminal de l'arbre en construction, on sélectionne le test qui génère le plus d'info sur les classes à expliquer/prédire dans chaque noeud créé

→ Critère à optimiser : Impureté :

- X ayant pour coordonnées les variables explicatives,
- 1, 2, ..., K les modalités de la variable Y à expliquer
- les modalités définissent K classes d'individus.

différence : Arbre de regression / Arbre de classif

critère de découpage d'un noeud

⇒ On définit la qualité d'un noeud m par l'impureté $i(m)$

A chaque noeud m → Région R_m

R_m : la/les indiv qui satisfont les conditions de tous les noeuds situés sur l'unique chemin reliant la racine à m

Chaque R_m contient N_m indiv

$$\hat{P}_{mK} = \frac{1}{N_m} \sum_{x_j \in R_m} \mathbb{1}_{(Y_i = K)}$$

→ Proportion d'indiv $\in K$ dans la région R_m

1 - $i(m)$ maximale → les indiv du noeud m se répartissent uniformément dans les classes, i.e. pour $\hat{P}_{mk} = \frac{1}{K}$ $\forall k \in \{1, ..., K\}$

→ $i(m)$ → minimale lorsque les individus de m appartiennent à la même classe

$$i(m) = \phi(\hat{p}_{mk}, \dots, \hat{p}_{mK})$$

$$\Delta i(m) = i(m) - \frac{N_{m_G}}{N_m} i(m_G) - \frac{N_{m_D}}{N_m} i(m_D)$$

m_G : nœud fils gauche de m

m_D : nœud fils droite de m

- $i_r(m)$: mesure le pourcentage d'individus classés incorrectement lorsque les individus d'un nœud sont affectés à sa classe majoritaire

$$\Delta i_r(m) = \frac{N_{m_G}}{N_m} \max_k (\hat{p}_{m_G k}) + \frac{N_{m_D}}{N_m} \max_k (\hat{p}_{m_D k}) - \max_k (\hat{p}_{mk})$$

Indice de gini : $i_g(m) = \sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$

Entropie : $i_e(m) = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$

Pour 2 classes

$$i_r(m) = 1 - \max(p, 1-p)$$

$$i_g(m) = 2p(1-p)$$

$$i_e(m) = -p \log(p) - (1-p) \log(1-p)$$

Règle d'affectation

$C_k = \sum_{k'=1}^{K} C_{kk'} \hat{p}_{mk'}$ ↦ coût d'affectation d'une feuille m à une classe k.

$C_{kk'}$: coût d'affectation d'un individu de k' dans k

$C(m) = \min_k C_k$ ↦ une feuille est affectée à la classe pour laquelle C_k est min.

Taux d'Erreur apparente de classement

A tout segment terminal m de l'arbre T associé à une classe K

$$\rightarrow \boxed{ER(K/m) = \sum_{K'=1,\ K \neq K'}^{K} \hat{p}_{mK'}}$$

$$TEA(T) = \sum_{m \in T} \frac{n_m}{n} ER(K/m)$$

↑ taux d'erreur apparente

Arbre de discrimination

- données ← read.table(file = "---", dec = ".", header = TRUE)
- donnees.cnt ← rpart.control(minsplit = 1)
  - donnees.cnt → stocke les param de l'algo
  - minsplit → nb min d'indiv nécessaire à la création d'un nœud
- Construire l'arbre sur la totalité des indiv (calculé) :

  arbre.full ← rpart(coeur~., data = donnees, method = "class", control = donnees.cnt)
- imprimer l'arbre : print( )
- sous forme graphique : plot(, uniform = TRUE, branch = 0.5, margin = 0.1)

  text(, all = FALSE, use.n = TRUE)
- Prédiction et Matrice de confusion :

```
pred ← predict(arbre.full, newdata = donnees, type = "class")
mc ← table(donnees$coeur, pred) #matrice de confusion
print(mc)
err.resub ← 1.0 - (mc[1,1] + mc[2,2])/sum(mc)
print(err.resub)   # erreur de resubstitution
```

- Elagage ; élaguer automatiquement un arbre avec la méthode

  prune(/pfit$arbre.full $cptable) → Arbre optimal

optimal :

- cp paramètre de complexité
- rel error (erreur d'entraînement)
- xerror mesure le taux d'erreur dans la validation croisée à 10 plis que l'on considère comme un estimateur correct de l'erreur réelle.
- xstd écart type de l'erreur de validation croisée

↓

l'arbre qui minimise xerror + xstd

3] élaguer arbre.full, indiquer la procédure à suivre pour obtenir un arbre optimal à partir de cet arbre.

Afin d'obtenir l'arbre optimal:

1- Consulter "CP Table" et plus précisément la somme des colonnes xerror et xstd pour évaluer l'erreur réelle

⟶ le meilleur arbre est celui associé à xerror + xstd minimal ⟶ nsplit = k donc k+1 feuilles

=> élaguer l'arbre actuel avec la commande prune en choisissant une valeur de CP entre]——, ——]

2) prune(arbre.full, cp=0,016)

5] Sélection pas à pas forward:

La sélection pas à pas se fait par:

minimisation du critère d'AKAIKE AIC = $-2LL + 2a(j+1)$

j: ns variable ~~qui minimise~~ déviance

à partir du modèle réduit à une constante puis de rajouter à chaque itération la variable qui minimise AIC.

La sélection s'arrête lorsque l'ajout d'une variable n'améliore plus le critère AIC