ECN6360 - Mégadonnées
6.Arbres de décision et forêts aléatoires

Chap 8 de JWHT et chap 9, 10, 15 de HTF

Marine Carrasco

Université de Montréal

UdeM

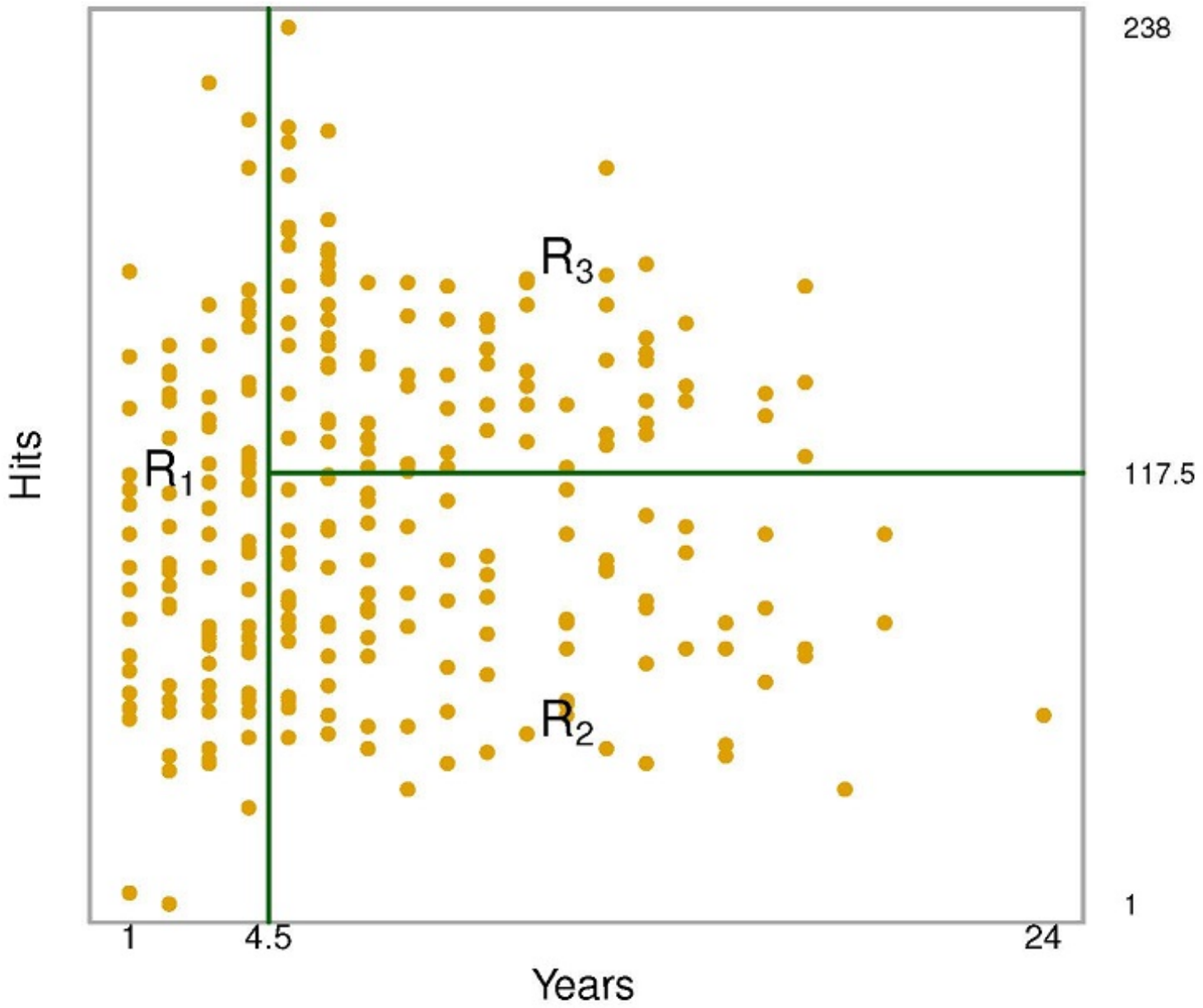
Méthodes basées sur les arbres

- Acronyme : CART (classification and regression tree)
- Méthodes utilisées pour la régression (y quantitative) et la classification (y qualitative).
- On partitionne l'espace des prédicteurs en plusieurs régions simples.
- Méthodes non paramétriques.
- L'algorithme nécessite
  - un critère permettant de sélectionner la meilleure division
  - un critère d'arrêt
  - une prédiction
- Les noeuds terminaux sont appelés feuilles et leur nombre correspond au nombre de régions.

Arbre quand Y est quantitative (arbre de régression)

Figure: Prédiction du salaire des joueurs de baseball basée sur *Years*, le nombre d'années jouées dans une league majeure, et *Hits*, le nombre de coups sûrs faits l'année précédente.

Régions pour l'arbre de régression



Terminologie

- Un arbre est toujours dessiné la tête en bas.
- Un arbre consiste à déterminer une séquence de noeuds (nodes).
  - Un noeud est défini par le choix conjoint d'une variable parmi les explicatives et d'une valeur seuil qui conduit à une division de l'échantillon en deux sous-groupes. La procédure est itérée sur chacun des sous-ensembles.
  - A la racine (ou noeud initial) correspond l'ensemble de l'échantillon.
  - Les segments d'arbres qui connectent deux noeuds sont appelés branches.

Prédiction avec un arbre

- On divise l'espace des prédicteurs (l'ensemble des valeurs de $X_1, ..., X_p$) en J régions distinctes non-superposées $R_1, ..., R_J$.
- Pour chaque observation qui tombe dans la région R_j, on fait la même prédiction : la moyenne de y dans l'échantillon d'entrainement dans cette région.
- Example avec une seule variable explicative X:
  - à l'étape 1, on sélectionne les régions $R_1(s)$ et $R_2(s)$
  - $R_1(s) = \{X : X < s\}$, $R_2(s) = \{X : X \geq s\}$.
  - Soit $\overline{y}_{R_1}$ et $\overline{y}_{R_2}$ les moyennes de y dans les deux régions $R_1(s)$ et $R_2(s)$.
  - Pour un s donné et X donné, la prédiction de y est soit $\overline{y}_{R_1}$, soit $\overline{y}_{R_2}$ selon que $X \in R_1(s)$ ou $X \in R_2(s)$.
  - Donc c'est équivalent à estimer le modèle à seuil suivant

$$y_i = \mu_1 I\{X_i < s\} + \mu_2 I\{X_i \geq s\} + \epsilon_i.$$

Exemple avec une variable explicative - choix du seuil s

- On veut choisir s qui minimise la somme des carrés de résidus.
- On ordonne les données X par ordre croissant.
- On fait une grille de valeurs de s de manière à ce qu'au moins 5% des données sont dans chaque classe.
- On choisit le s de cette grille qui minimise

$$MSE\,(s) = \sum_{i:X_i \epsilon R_1(s)} \left(y_i - \overline{y}_{R_1(s)}\right)^2 + \sum_{i:X_i \epsilon R_2(s)} \left(y_i - \overline{y}_{R_2(s)}\right)^2$$

$\Rightarrow \widehat{s}$.

Exemple avec une variable explicative - faire pousser l'arbre

- Une fois que l'on a $\widehat{s}$, on a les régions R_1 et R_2.
- On divise la région R_1 en deux régions et on choisit la division qui minimise la somme des carrés des résidus (dans la région R_1).
- On divise la région R_2 en deux régions et on choisit la division qui minimise la somme des carrés des résidus (dans la région R_2).
- On continue à diviser les régions successivement.
- Chaque division va réduire la SCR.
- On s'arrête lorsque chaque feuille a moins qu'un certain nombre d'observations, par exemple 5 ou 10.
- Un arbre complet risque de sur-ajuster les données, il va falloir élaguer.

Cas avec plusieurs variables explicatives - faire pousser l'arbre

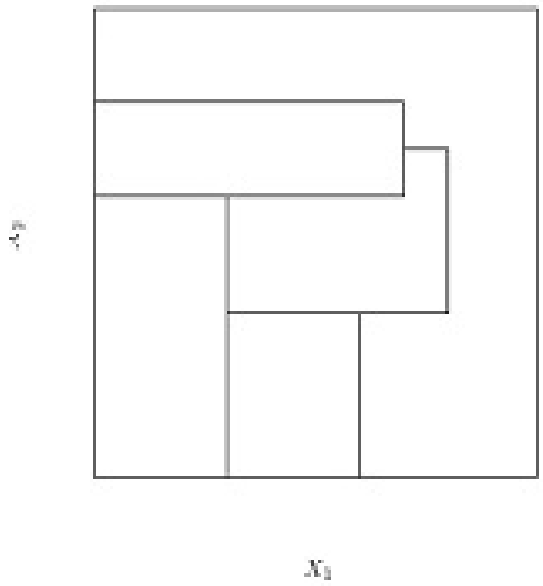
- Supposons que l'on a p variables explicatives.
- A la $1^{ère}$ étape, on choisit la **variable explicative** j et le **seuil** s qui minimise

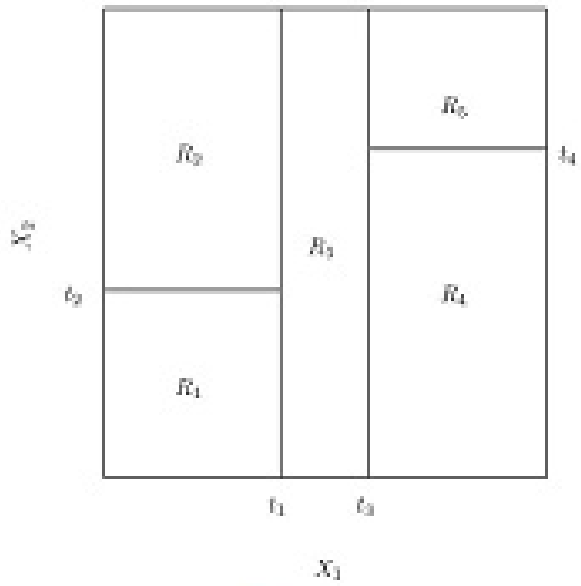
$$\sum_{i:X_{ij}\epsilon R_1(j,s)} \left(y_i - \overline{y}_{R_1}\right)^2 + \sum_{i:X_{ij}\epsilon R_2(j,s)} \left(y_i - \overline{y}_{R_2}\right)^2$$

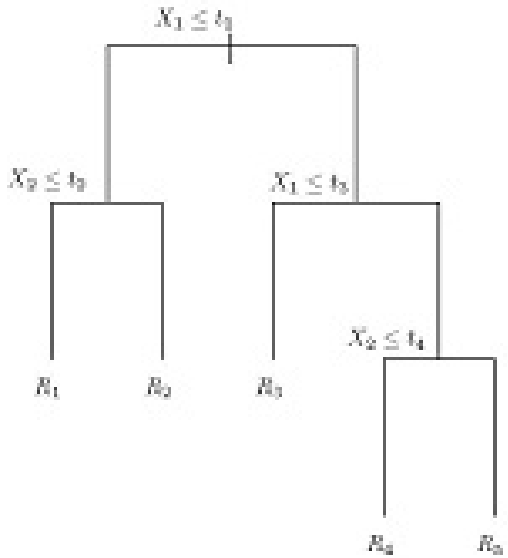
où $R_1(j,s)$ est la région où X_j est telle que $X_j < s$ et $R_2(j,s)$ est la région où X_j est telle que $X_j \geq s$ et $\overline{y}_{R_1}$ et $\overline{y}_{R_2}$ sont les moyennes de y dans chacune de ces régions.

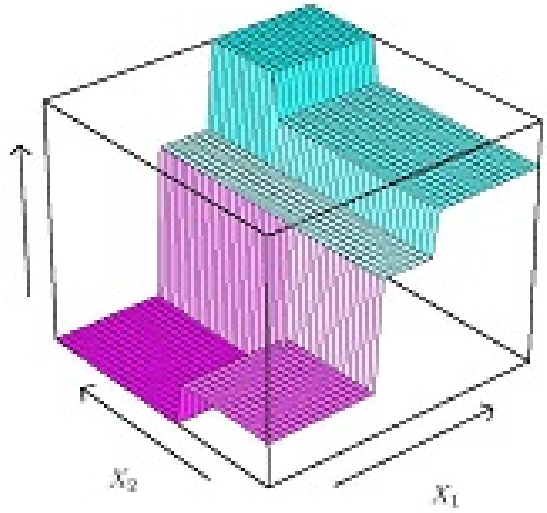
- À la $2^{ème}$ étape, on sépare chaque région R_1 et R_2 de manière à obtenir la plus grande réduction de la SCR, on obtient alors 4 régions.
- On procède ainsi séquentiellement. On parle de partitionnement récursif.

Exemples d'arbres et de régions









Elagage (tree pruning)

- La construction d'un arbre complet peut conduire à une division trop raffinée et donc à un modèle de prévision très instable car fortement dépendant de l'échantillon qui a permis l'estimation. La prévision hors échantillon aura alors un grand MSE.
- C'est une situation de sur-ajustement à éviter au profit d'un modèle plus parcimonieux qui donnera de meilleures prévisions.
- On procède à l'élagage: élimination de branches pour obtenir un sous-arbre. On peut évaluer la performance d'un sous-arbre par validation croisée.
- Tous les sous-arbres sont admissibles mais, comme leur nombre croît de manière exponentielle, il n'est pas envisageable de tous les considérer.
- Solution : construire une suite emboitée de sous-arbres et choisir, parmi cette suite seulement, l'arbre qui minimise un critère.

Construction d'une suite d'arbres par pénalisation de la complexité

- Pour un arbre donné T, on note $|T|$ le nombre de feuilles ou noeuds terminaux. $|T|$ mesure la complexité de l'arbre.
- Chaque noeud terminal m correspond à une région R_m.
- Soit l'arbre complet T_0. On construit une suite de sous-arbres indexés par un paramètre de pénalisation α.
- A chaque valeur de α, on choisit le sous-arbre $T_\alpha \subset T_0$ tel que

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \widehat{y}_{R_m})^2 + \alpha |T|$$

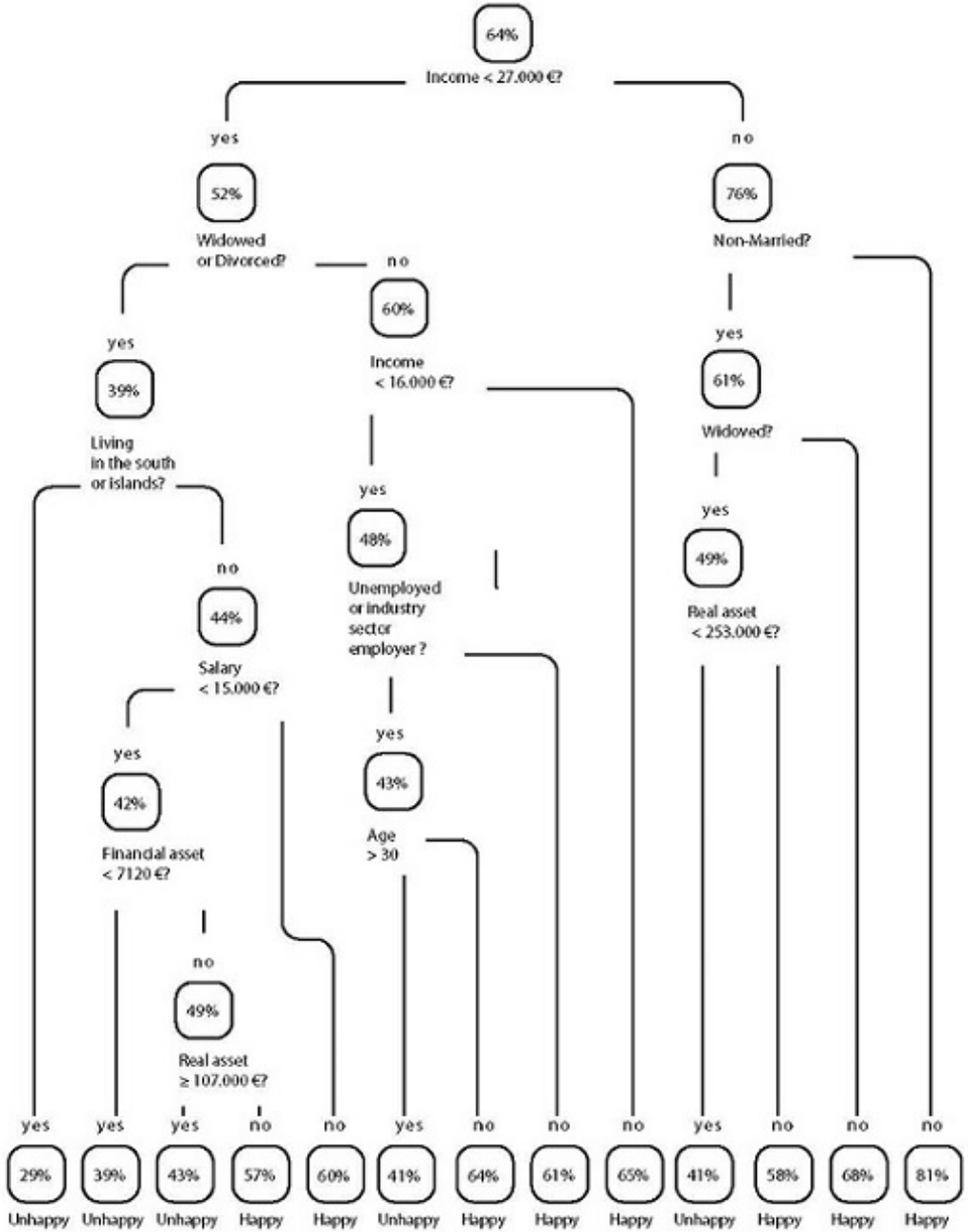
  est minimal. $\widehat{y}_{R_m}$ est la valeur prédite associée à R_m (c'est-à-dire la moyenne des observations dans R_m).
- Le procédé est itéré pour la construction de la séquence emboitée de sous-arbres.

- Pour $\alpha = 0$, le sous-arbre est simplement $T_0 \Rightarrow$ biais faible, variance grande.
- Quand α augmente, la pénalisation sur le nombre de noeuds terminaux conduit à retenir des arbres de plus en plus petits (des branches tombent) $\Rightarrow$ biais plus grand, variance plus petite.
- α est choisi par validation croisée, typiquement 5 ou 10-fold CV $\Rightarrow \widehat{\alpha}$.
- L'arbre final est $T_{\widehat{\alpha}}$.

Algorithme pour la construction de l'arbre de régression

1. Faire pousser un grand arbre en s'arrêtant quand le nombre d'observations par région est $\leq$ à une certaine valeur.
2. Élaguer l'arbre en pénalisant sa complexité pour obtenir une suite de sous-arbres indexés par α.
3. Utiliser K-fold cross-validation pour sélectionner α. C'est-à-dire diviser l'échantillon d'entrainement en K sous-échantillons.
   (a) Pour chaque $k = 1, 2, ..., K$: répéter l'étape 1 et 2 pour l'ensemble des données sauf celles du $k$ème échantillon.
   (b) Pour chaque α, évaluer l'EQMP sur les données du $k$ème sous-échantillon. Prendre la moyenne des EQMP pour les K sous-échantillons. Sélectionner le α qui minimise la moyenne des EQMP.
4. Retourner le sous-arbre de l'étape 2 qui correspond à la valeur choisie de α.

Déterminants du bonheur (S. Galletta)



Arbre quand Y est qualitative (arbre de classification)

- Très similaire à l'arbre de régression mais on prédit une variable qualitative.
- Dans l'arbre de régression, la prévision est la moyenne sur une région.
- Ici, on utilise la classe la plus fréquente dans l'échantillon d'entrainement pour la région dans laquelle l'observation appartient.
- Comme pour l'arbre de régression, on fait pousser l'arbre par divisions successives.

- Quand Y est une variable quantitative, on prend la division qui minimise

$$\sum_{m=1}^{M} N_m Q_m$$

où $Q_m = \frac{\sum_{i:x_i \in R_m} (y_i - \widehat{y}_{R_m})^2}{N_m}$ et $N_m =$ nombre d'observations x_i dans R_m.

- Comme maintenant Y est une variable qualitative, on ne peut pas utiliser le MSE pour trouver la meilleure division.
- On pourrait remplacer Q_m par l'erreur de classification E_m,
- mais on préfère utiliser l'indice de Gini G_m ou l'entropie D_m.

Erreur de classification

- Soit $\widehat{p}_{mk}$ la proportion d'observations d'entrainement de la $m^{ème}$ région qui appartiennent à la classe k :

$$\widehat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I\left(y_i \in k\right).$$

- Soit $k(m)$ la classe la plus probable : $k\left(m\right) = \arg\max_k \widehat{p}_{mk}$.
- L'erreur de classification de la $m^{ème}$ région est

$$E_m = \frac{1}{N_m} \sum_{x_i \in R_m} I\left(y_i \notin k\left(m\right)\right) = 1 - \widehat{p}_{mk(m)} = 1 - \max_k \widehat{p}_{mk}.$$

- L'erreur de classification est $E = \sum_{m=1}^{M} \frac{N_m}{N} E_m$.
- Choisir les divisions en minimisant l'erreur de classification conduit à des régions insuffisamment pures (qui contiennent trop d'observations de différentes classes).
- À la place, on utilise l'indice de Gini ou l'entropie.

Indice de Gini

- L'indice de Gini pour la $m^{ème}$ région

$$G_m = \sum_{k=1}^{K} \widehat{p}_{mk} \left(1 - \widehat{p}_{mk}\right).$$

- $G_m \in [0, 1]$. L'indice de Gini est petit quand les $\widehat{p}_{mk}$ sont proches de 0 ou 1.
- L'indice de Gini est une mesure de la variance à travers les K classes. Il mesure aussi la pureté de la classe. Une petite valeur indique que la région comprend principalement des observations d'une même classe.
- On choisit la division qui donne le plus petit indice de Gini :

$$G = \sum_{m=1}^{M} \frac{N_m}{N} G_m.$$

Entropie croisée

- L'entropie croisée de la $m^{ème}$ région

$$D_m = -\sum_{k=1}^{K} \widehat{p}_{mk} \ln \widehat{p}_{mk}$$

- Comme $0 \leq \widehat{p}_{mk} \leq 1$, il en résulte que $-\sum_{k=1}^{K} \widehat{p}_{mk} \ln \widehat{p}_{mk} \geq 0$.

  On utilise la convention $0 \ln(0) = 0$.
- L'entropie est proche de 0 quand $\widehat{p}_{mk}$ est proche de zéro ou de 1.
- Donc comme l'indice de Gini, l'entropie prendra une petite valeur quand la $m^{ème}$ région est pure. Similaire numériquement à l'indice de Gini.
- On choisit la division qui donne la plus petite entropie croisée :

$$D = \sum_{m=1}^{M} \frac{N_m}{N} D_m.$$

Elagage

- Pour chaque valeur de α, on sélectionne le sous-arbre T_α qui minimise

$$\sum_{m=1}^{|T|} N_m Q_m + \alpha |T|$$

où $|T|$ = nombre de noeuds terminaux et N_m = nombre d'observations x_i dans R_m.

- Pour Q_m, on peut utiliser soit l'erreur de classification E_m, soit l'indice de Gini G_m ou l'entropie D_m.
- Si le but est la précision de la prédiction, alors il est préférable d'utiliser l'erreur de classification.
- On choisit α par validation croisée.

Comparaison entre arbre et régression linéaire

Dans la régression linéaire, le modèle est

$$y = \beta_0 + \sum_{j=1}^{p} \beta_j X_j + \varepsilon$$

alors que dans l'arbre de régression, on suppose

$$y = \sum_{m=1}^{M} c_m I_{(X \in R_m)} + \varepsilon$$

où $R_1,...,R_m$ représentent un partitionnement de l'espace des variables explicatives.
Quel modèle est le meilleur?

- Si le vrai modèle est linéaire alors la régression linéaire sera meilleure.
- Si le vrai modèle est très non-linéaire, alors l'arbre de régression sera meilleur.
- L'arbre de régression est une méthode non paramétrique qui impose peu de contraintes sur le modèle.

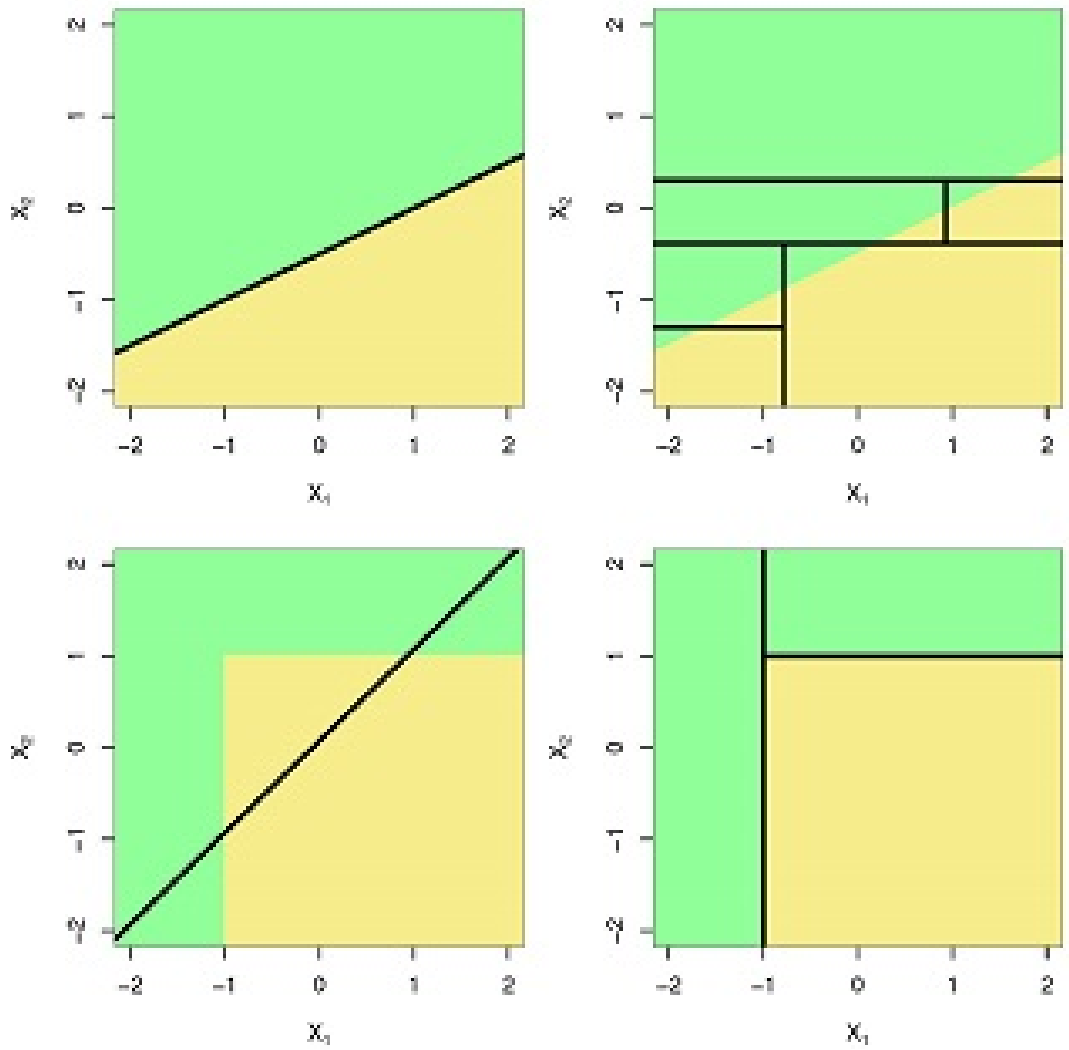


Figure: Classification. Haut : cas où la frontière de décision est linéaire et représentée par les zones vertes et jaunes. Gauche: classification linéaire (ex : logit), droite : arbre. Bas : Cas non-linéaire.

Avantages des arbres

- Les arbres sont faciles à expliquer, même à des non-spécialistes.
- Les arbres peuvent être représentés graphiquement et sont facilement interprétables.
- Ils s'appliquent à la fois aux variables quantitatives et qualitatives sans avoir besoin de créer des variables binaires.
- Pas besoin de standardiser les données.
- Peu sensibles aux valeurs extrêmes.
- S'applique à des modèles très non linéaires.

Désavantages des arbres

- Leur prédiction n'est pas toujours très bonne.
- Ils tendent à être instables, un petit changement dans les données peut résulter en un grand changement dans l'arbre estimé.

Pour améliorer la prévision et réduire la variance, des méthodes d'agrégation de plusieurs arbres sont utilisées : bagging, forêts aléatoires, boosting.

Bagging

Bagging = **B**ootstrap **Agg**regation

- En général, on utilise le bootstrap quand c'est difficile ou impossible de calculer une quantité à laquelle on est intéressé telle que l'écart-type.
- Ici, on l'utilise pour une différente raison.
- Les arbres complets ont une grande variance.
- Bagging est une méthode générale pour réduire la variance d'une méthode d'apprentissage en faisant la moyenne de plusieurs prédictions.
- Intuition : Si $Z_1, ..., Z_n$ sont n variables indépendantes de variance σ^2, alors leur moyenne a une variance égale à σ^2 / n. Donc, prendre la moyenne réduit la variance.

Bagging pour les arbres de régression

- Idée : tirer B échantillons bootstrap dans l'échantillon initial (tirage avec remise), faire pousser un arbre complet pour chaque échantillon bootstrap b. Ne pas élaguer. Les arbres complets ont un biais petit mais une grande variance.
- Calculer la prédiction pour chaque échantillon bootstrap $b = 1, 2, ..., B : \widehat{f}_1^* (x), ..., \widehat{f}_b^* (x), ..., \widehat{f}_B^* (x)$, ces prédictions sont des moyennes pour une région R_m telle que $x \in R_m$.
- Ensuite, on prend la moyenne de ces prédictions

$$\widehat{f}_{avg} (x) = \frac{1}{B} \sum_{b=1}^{B} \widehat{f}_b^* (x) .$$

- Remarques : Comme on fait la moyenne d'échantillons bootstrap, on parle de **bagging**. La méthode n'est pas très sensible au choix de B. Il suffit de le prendre suffisamment grand, par exemple $B = 500$.

Bagging pour les arbres de classification

- Calculer la classe prédite pour chaque échantillon bootstrap, $b = 1, 2, ..., B$.
- Ensuite, utiliser la règle de la majorité: la prédiction est la classe qui apparaît le plus souvent parmi les B prédictions.
- Par exemple, s'il y a deux classes : **défaut** et **non-défaut** et que pour une région de x plus de 50% des arbres bootstrap prédisent **défaut**, alors la prédiction bagging sera **défaut** pour ce x.

Evaluation de la performance : out-of-bag error

Comment calculer l'erreur test pour bagging?

- Pas besoin d'utiliser la validation croisée car bagging offre une méthode très simple pour évaluer l'erreur test.
- L'échantillon initial est de taille n mais certaines observations sont omises de chaque échantillon bootstrap (car tirage avec remise).
- La chance qu'une observation ne soit pas dans l'échantillon bootstrap est

$$\left(1 - \frac{1}{n}\right)^n \sim e^{-1} \sim \frac{1}{2,73} \sim \frac{1}{3}.$$

- Donc chaque arbre b utilise environ $2/3$ de l'échantillon,
- les autres observations sont les observations "out-of-bag",
- on peut s'en servir comme observations tests.

Estimation de l'erreur par out-of-bag (OOB)

- Prédire la réponse pour la $i^{ème}$ observation en utilisant la prédiction moyenne (ou la règle de la majorité) pour tous les arbres pour lesquels cette observation est OOB (c'àd n'est pas utilisée dans l'échantillon bootstrap).
- Répéter ceci pour chaque observation et calculer le MSE OOB ou l'erreur de classification OOB.
- Ceci représente un estimateur valide de l'erreur test si B est suffisamment grand, car c'est proche du LOOCV.
- **Désavantages du bagging** : La variance de cet estimateur peut être encore grande car les arbres sont corrélés.
- L'estimateur bagging est moins facile à interpréter car on fait la moyenne sur plusieurs arbres. Pour illustrer l'importance des différentes variables explicatives, on peut reporter la réduction de la SCR (pour l'arbre de régression) ou de l'indice de Gini (pour l'arbre de classification) qui résulte de la division d'une variable explicative. Plus la réduction est grande, plus la variable est importante.

Réduction de variance résultant du bagging

- Les arbres $\widehat{f}_b^*$ obtenus pour les différents échantillons bootstrap sont identiquement distribués donc le biais $\widehat{f}_{avg}$ sera le même que celui de chaque arbre individuel.
- Si les arbres $\widehat{f}_b^*$ étaient indépendants alors $V\left(\widehat{f}_{avg}\right) = \frac{V\left(\widehat{f}_b^*\right)}{B} \equiv \frac{\sigma^2}{B}$.
- Toutefois les arbres sont corrélés entre eux car ils utilisent presque les mêmes données et toutes les variables explicatives.
- Si les $\widehat{f}_b^*$ ont une corrélation ρ, on peut montrer que $V\left(\widehat{f}_{avg}\right) = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$. Donc $V\left(\widehat{f}_{avg}\right) \to \rho\sigma^2 > 0$ quand $B \to \infty$.
- Comme les arbres sont fortement corrélés, la réduction de variance n'est pas très grande. Un solution est de réduire la corrélation entre les arbres. C'est le principe utilisé par les forêts aléatoires.

Forêts aléatoires

- Dans le bagging, on utilise à chaque division l'ensemble des variables explicatives disponibles, donc les arbres construits sont très similaires.
- Donc les arbres sont fortement corrélés et prendre leur moyenne ne résulte pas en une grande réduction de la variance.
- Pour éviter ceci, on construit ce qu'on appelle "une forêt aléatoire".
- Quand on construit l'arbre, à chaque division, on prend un sous-ensemble de variables explicatives (par défaut $m = \sqrt{p}$).
- Ceci permet d'obtenir des arbres plus différents les uns des autres, ce qui conduit à "décorréler" les arbres.

Algorithme : forêt aléatoire pour régression et classification

1. Pour $b = 1$ à B :
    1. Tirer un échantillon bootstrap dans les données d'entrainement.
    2. Faire pousser un arbre de forêt aléatoire T_b à partir des données bootstrap en répétant les étapes suivantes pour chaque noeud terminal de l'arbre jusqu'à ce que le nombre minimal d'observations par noeud atteigne une valeur $n_{\min}$.
        - (i) Sélectionner m variables au hasard parmi les p prédicteurs.
        - (ii) Choisir la meilleure division (j, s) parmi les m.
        - (iii) Séparer la région en deux sous-régions.
2. Retourner l'ensemble des arbres $\widehat{f}_b$, $b = 1, 2, ..., B$.

- Pour faire une **prédiction** en un point x :
  - Régression :
$$\widehat{f}_{rf} = \frac{1}{B} \sum_{b=1}^{B} \widehat{f}_b(x).$$
  - Classification : Soit $\widehat{C}_b(x)$ la classe prédite pour le $b^{ème}$ arbre de forêt aléatoire. Alors $\widehat{C}_{rf}(x)$ est la classe la plus fréquemment prédite parmi les B prédictions.
- L'erreur test peut être calculée par out-of-bag comme pour bagging.

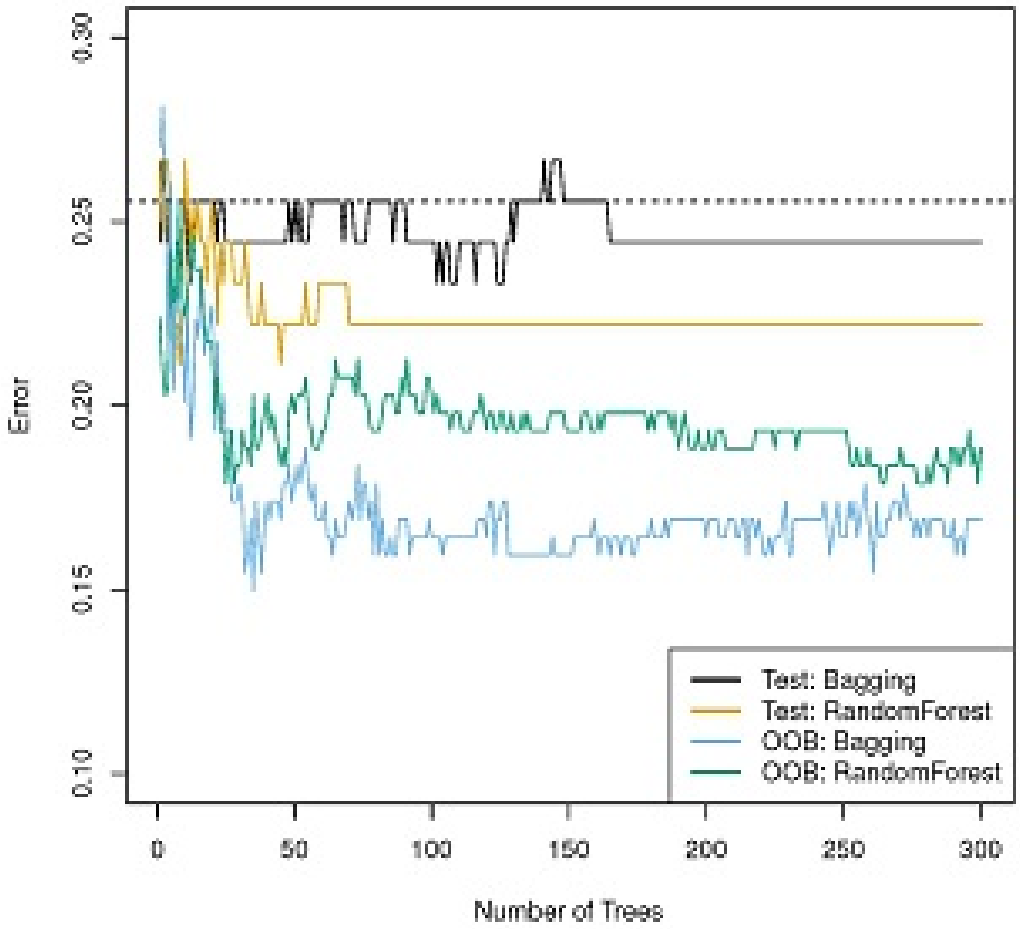


Figure: L'erreur test est montrée comme fonction de B. On voit que la forêt aléatoire a une erreur test plus faible que bagging.

Boosting

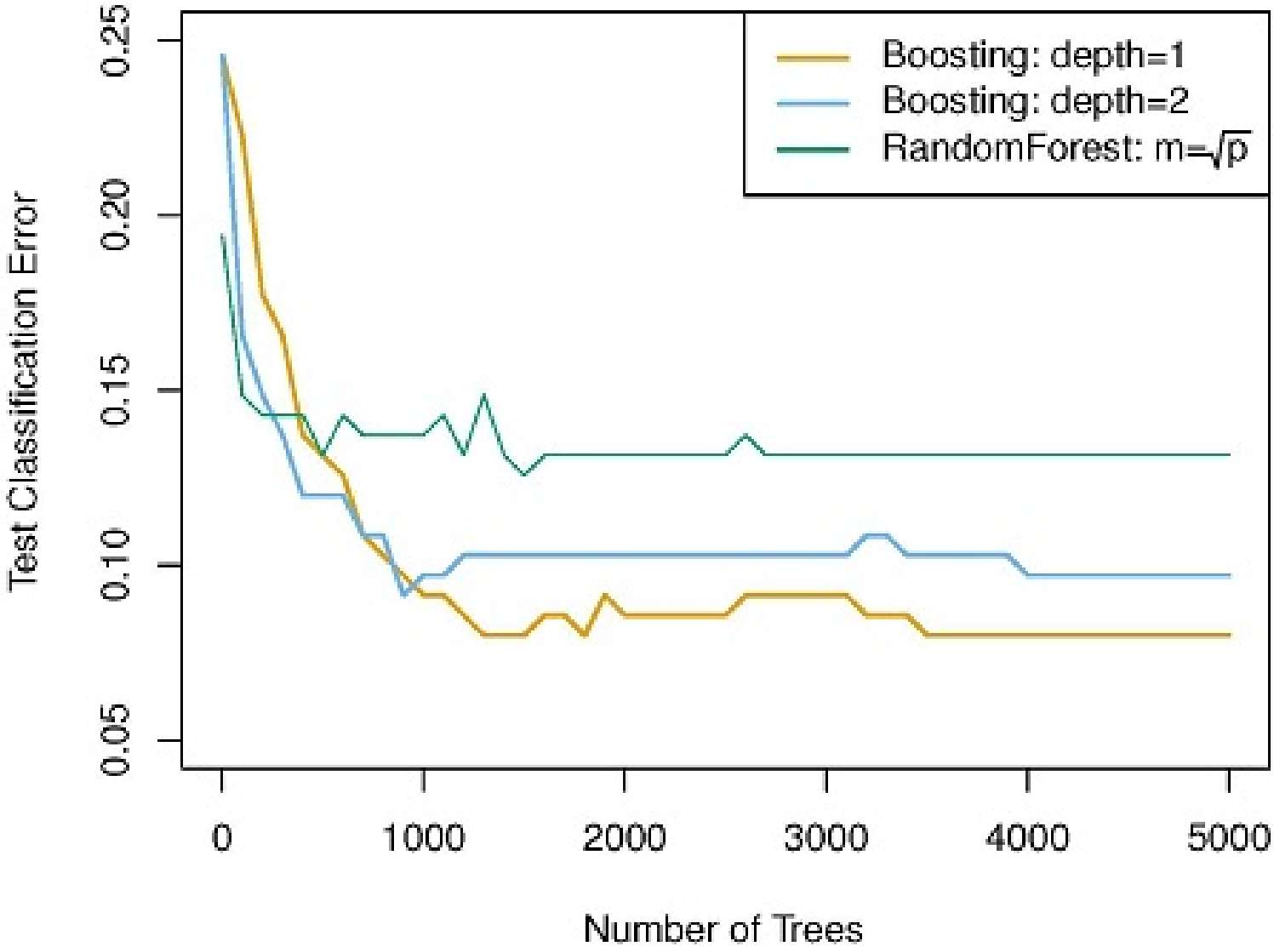
- Méthode pouvant s'appliquer à beaucoup de méthodes d'apprentissage. Ici on présente cette méthode dans le contexte des arbres de décision.
- Boosting est une approche alternative pour améliorer la prévision des arbres et n'utilise pas le bootstrap.
- Cette méthode fait grandir lentement un grand nombre de petits arbres $\widehat{f}_1(x), ..., \widehat{f}_B(x)$ séquentiellement.
- A chaque étape, on construit un arbre sur les résidus du modèle précédent.

Algorithme du boosting pour les arbres

1. Poser $\widehat{f}(x) = 0$ et $r_i = y_i$ pour tous les i dans l'échantillon d'entrainement.
2. Pour $b = 1, 2, ..., B$, répéter :
   (a) Construire un arbre $\widehat{f}^b$ avec d divisions ($d + 1$ noeuds terminaux) pour prédire r_i avec x_i.
   (b) Mettre à jour $\widehat{f}$ en ajoutant une version pénalisée du nouvel arbre,
   $$\widehat{f}(x) \leftarrow \widehat{f}(x) + \lambda \widehat{f}^b(x).$$
   (c) Mettre à jour les résidus,
   $$r_i \leftarrow r_i - \lambda \widehat{f}^b(x_i).$$
3. Prédiction obtenue par boosting,
$$\widehat{f}(x) = \sum_{b=1}^{B} \lambda \widehat{f}^b(x).$$

Choix des paramètres de lissage dans le boosting

- Le nombre d'arbres B : Contrairement au bagging et forêts aléatoires, le boosting risque de sur-ajuster si B est trop grand. On peut le sélectionner par validation-croisée.
- La paramètre de pénalisation λ contrôle la vitesse d'apprentissage, $0 < \lambda < 1$. Le choisir petit, par ex : 0,01 ou 0,001.
- Le nombre de divisions d contrôle la complexité. Le prendre petit, souvent $d = 1$ marche bien.
- d est appelé "interaction depth".


Boosting: depth=1
Boosting: depth=2
RandomForest: m=√p
Test Classification Error
0.05
0.10
0.15
0.20
0.25
0
1000
2000
3000
4000
5000
Number of Trees

Application à la détection de spam

- 4601 messages
- y variable binaire = email (codé 0) ou spam (codé 1).
- Variables explicatives:
  - 48 variables : % d'occurences d'un mot particulier (ex: business, address, internet, free, george) dans le message.
  - 6 variables correspondant au nombre de fois où un certain caractère apparait (ch;, ch(, ch[, ch!, ch$, ch#).
  - longueur moyenne de suites ininterrompues de lettres majuscules CAPAVE.
  - longueur de la plus longue suite ininterrompue de lettres majuscules CAPMAX.
  - somme des longueurs des suites ininterrompues de lettres majuscules CAPTOT.
- On partage l'échantillon en un échantillon test de 1536 données prises au hasard et un échantillon d'entrainement de 3065 données.

Modèle logistique additif généralisé pour classification avec spline cubique de lissage avec degré de liberté $= 4$.

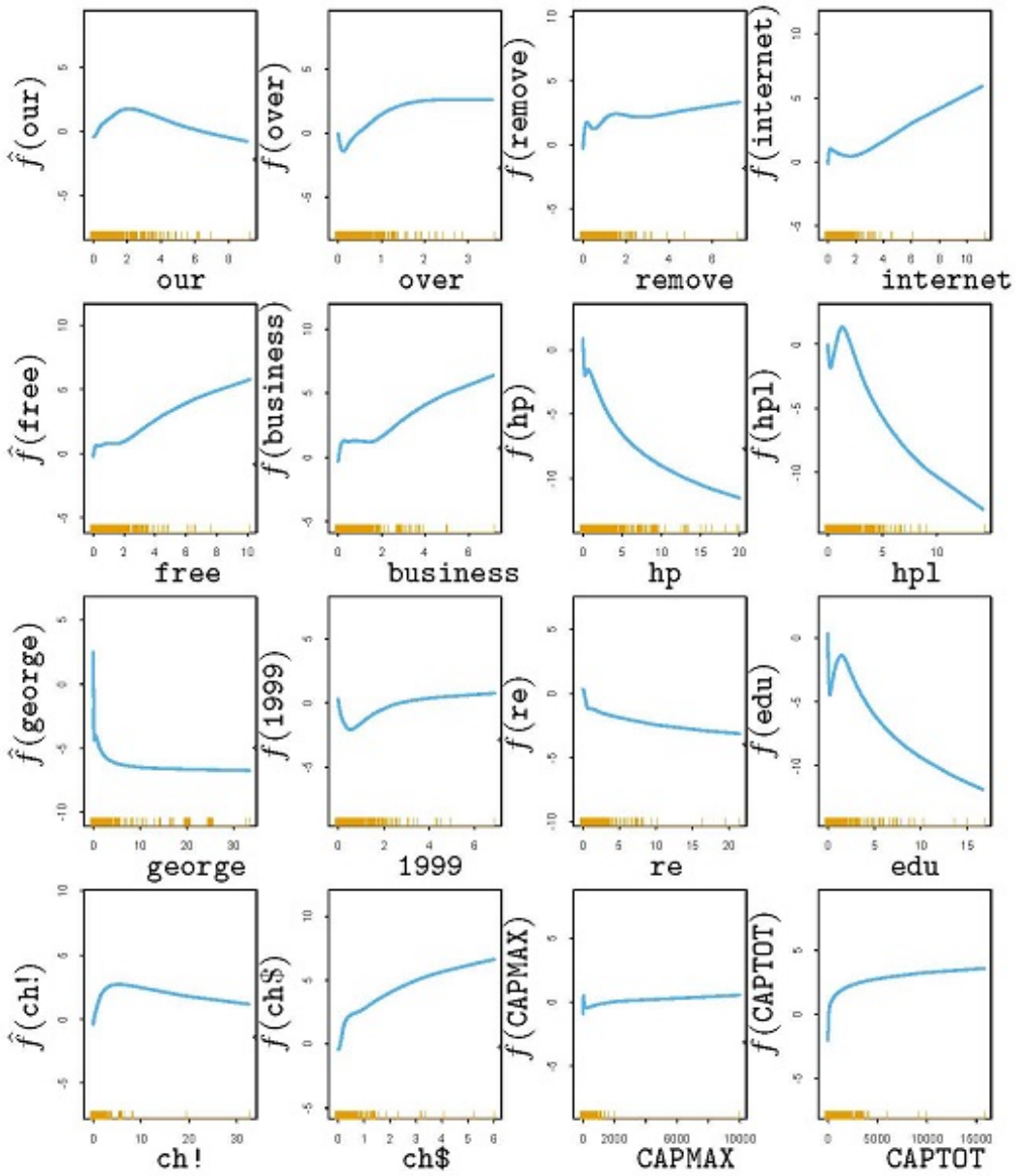
$$\ln\left(\frac{P\left(Y=1|X\right)}{P\left(Y=0|X\right)}\right)=\beta+f_1\left(X_1\right)+...+f_p\left(X_p\right).$$

Matrice de confusion

	classe prédite	
vraie	email (0)	spam (1)
email (0)	58,3%	2,5%
spam (1)	3%	36,3%

Taux d'erreur global $=$ 5,5%

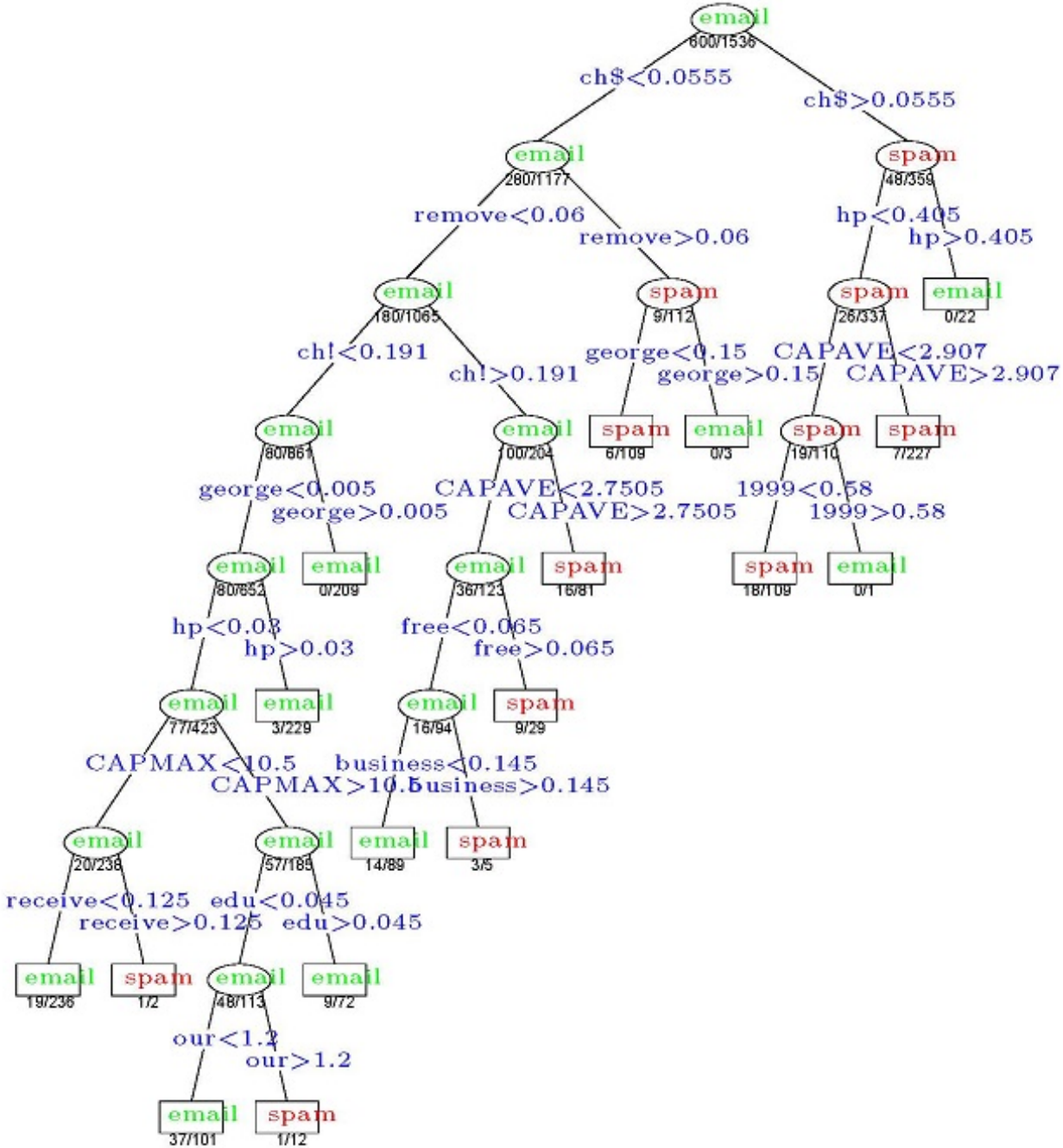
- Variables qui ont un effet positif sur la probabilité d'être un spam : our, over, remove, internet, free, business, ch!, ch$, CAPMAX, CAPTOT.
- Variables qui ont un effet négatif sur la probabilité d'être un spam : hp, george, 1999, edu.


$\hat{f}$(our)
our
f(over)
over
f(remove)
remove
f(internet)
internet
$\hat{f}$(free)
free
f(business)
business
f(hp)
hp
f(hpl)
hpl
$\hat{f}$(george)
george
f(1999)
1999
f(re)
re
f(edu)
edu
$\hat{f}$(ch!)
ch!
f(ch$)
ch$
f(CAPMAX)
CAPMAX
f(CAPTOT)
CAPTOT

Arbre de décision

- On a fait pousser un grand arbre puis on l'a élagué en utilisant le taux d'erreur de classification.
- α est sélectionné par 10-fold cross validation, ce qui donne un arbre à 17 noeuds terminaux.
- Taux d'erreur global est 9,3%.
- Matrice de confusion :

	classe prédite	
vraie	email (0)	spam (1)
email (0)	57, 3%	4%
spam (1)	5, 3%	33, 4%


email
600/1536
ch$<0.0555
ch$>0.0555
email
280/1177
spam
48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email
180/1065
spam
9/112
spam
26/337
email
0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email
80/861
email
100/204
spam
6/109
email
0/3
spam
19/110
spam
7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email
80/652
email
0/209
email
36/123
spam
15/81
spam
18/109
email
0/1
hp<0.03
hp>0.03
free<0.065
free>0.065
email
77/423
email
3/229
email
16/94
spam
9/29
CAPMAX<10.5
CAPMAX>10.5
business<0.145
business>0.145
email
20/238
email
57/185
email
14/89
spam
3/5
receive<0.125
receive>0.125
edu<0.045
edu>0.045
email
19/236
spam
1/2
email
48/113
email
9/72
our<1.2
our>1.2
email
37/101
spam
1/12