I. Introduction

- $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ local loss function

On veut résoudre : $\arg\min_f \mathbb{E}_{(X,Y)\sim D} [\ell(Y, f(X))]$

- Classification :

  (0/1) prediction loss $\to$ best classifier is Bayes

  $$f_{Bayes}(x) = \operatorname{argmax}_c \mathbb{P}(Y = c \mid x)$$

- Regression :

  ℓ_2 loss : $(y - f(x))^2 \to$ best solution in regression is

  $$f_{reg}(x) = \mathbb{E}(Y \mid x)$$

- Min of empirical risk : $\operatorname{argmin}_{f \in \mathcal{H}} \frac{1}{n} \sum_i^n \ell(y_i, f(x_i)) + \lambda \Omega(f)$

- Statistical learning : entrée : $\mathcal{H}, \ell, \Omega, S_n$

  sortie : $f_n = \underbrace{\mathcal{A}}_{\text{learning algo}} (\underbrace{S_n}_{\text{train data}}, \underbrace{\mathcal{H}}_{\text{class of } f}, \underbrace{\ell}_{\text{loss } f}, \underbrace{\Omega}_{\text{complexity measure}})$

Parametric VS non Parametric
↳ nb of param ↗ with size of training

Tree-based : ✓ transparency, pre-training, interprétabilité
✗ dépend bcp du nb de training data

II. Decision and regression tree

A. Predictive models

Decision trees :

- model of the form : $f(x) = \sum_{\ell=1}^{m} \mathbb{1}(x \in R_\ell) f_\ell$

$\rightarrow$ fonctions en escalier

$f_\ell = \arg\max_c \hat{p}_{\ell c}$ avec $\hat{p}_{\ell c} = \frac{1}{N_\ell} \sum_{x_i} \mathbb{1}(y_i = c)$

- make partition based on local criteria
- orthogonal separators
- leaf = majority vote

Regression trees :

model of the form : $f(x) = \sum_{\ell=1}^{M} \mathbb{1}(x \in R_\ell) f_\ell$

avec $f_\ell = \frac{1}{N_\ell} \sum y_i$

- leaf = local average

B. Learning trees

continuous: $t_{j,s}(x) = \text{sign}(x^j - s)$

feature $\uparrow$ $\uparrow$ seuil

categorical: $t_{j,v,k}(x) = \mathbb{1}(x^j = v_k^j)$

Recursive building algo :

- Build root node
- Current node $\rightarrow$ find best binary separation t
- Apply t to the current node and split in left & right nodes
- Compute stopping criterion on right . satisfied? $\xrightarrow{\text{Yes}}$ leaf
- Compute stopping criterion on left satisfied? $\xrightarrow{\text{Yes}}$ leaf

No

No

Best split for regression ?

2 plans : $\mathcal{R}_l(j,s) = \{x,\ x^j \leq s\}$
$\mathcal{R}_r(j,s) = \{x,\ x^j > s\}$

→ on veut minimiser $L((j,s),S)$ cad :

$$\min_{j,s} \left[\sum_{x_i \in S \cap \mathcal{R}_r} (y_i - \hat{y}_r)^2 + \sum_{x_i \in S \cap \mathcal{R}_l} (y_i - \hat{y}_l)^2 \right]$$

moy des y_i à d   moy des y_i à g

→ en pratique il suffit de minimiser la variance empirique

$$\min_{j,s} N_r \, Var(Y \mid S \cap \mathcal{R}_r(j,s)) + N_l \, Var(Y \mid \mathcal{R}_l(j,s))$$

Best split for classification ?

goal = minimize an impurity criterion

$$\min \left[\frac{N_r}{N} H(S \cap \mathcal{R}_r(j,s)) + \frac{N_l}{N} H(S \cap \mathcal{R}_l(j,s)) \right]$$

Impurity criteria : $p_k(S) = \frac{1}{m} \sum_{1}^{m} \mathbb{1}(y_i = k)$ → proba de chaque classe

- Cross-entropy : $H(S) = -\sum_{k=1}^{C} p_k(S) \log p_k(S)$
- Gini : $H(S) = \sum_{k=1}^{C} p_k(S)(1 - p_k(S))$
- Missclassification : $H(S) = 1 - p_{k(S)}$

Stopping criterion : • maximal depth
• un noeud a 1 nb min de data

améliorable par cross-validation

Désavantage : très variable en fonction des données
→ 1 petite variat° entraine un arbre complètement différent

III_ Introduction to ensemble methods

How to combine models ?

Ensemble Method	Committee Machines (Wisdom of the crowd)	Mixture of Experts
Same Base Model Combinat° linear eg: Bagging, Rd Forest, Boosting	Diverse base model Combination can be non-linear ↳ learned eg: Blending, Stacking	Models are experts for a certain data type On use le bon modèle en fonct° de la data

Focus on Ensemble methods

→ bootstrap samples (Bagging & Random Forest)
→ randomized pred (Rd. F)
→ weighted version of sample (Boosting)

IV_ Bagging

Reminder : noise = error by the Bayes model

biais = diff btw minimal error (Bayes error) and the avg model

variance = how much h_S varies from 1 training set to another

Algo: ① draw T bootstrap samples $\{B_1, ..., B_T\}$ from S
uniform "sampling with replacement

② $\forall B_T$, learn a model f_t

③ Build avg model : $f_{bag}(x) = \frac{1}{T} \sum_T f_t(x)$

Useful when learning algo unstable with a high variance
→ genre les arbres à tout hasard
↳ rarement utilisé seul
↳ pour classifier, un mauvais modèle empire la situation

VI - Random Forests

RF algo : • Boucle de 1 à T :

① Bootstrap sample from S

② Build randomized DT $h^{(t)}_{tree}$

/!\ do not prune

• at each node :

① select k features sans remise

② choose the cut

out : $H^T(x) = \frac{1}{T} \sum h^{(t)}_{tree}(x)$

Extra trees algo : • Boucle de 1 à T

① Prend S_{train} en entier

② Build randomized DT $h^{(t)}_{tree}$

• a each node :

① select k features sans remise

② Draw k splits using Pick-a-rd-split

• a^i_{max}, a^i_{min} extrema of x^i

• draw cut point in $[a^i_{max}, a^i_{min}]$

③ Choose the best split ds les k

out : $H^T(x) = \frac{1}{T} \sum h^{(t)}_{tree}(x)$

RF pros & cons

X - if tree too large → overfitting
- perte d'interprétabilité

V - rapide et marche bien quand ya bcp de features
- facile à tune

Variable importance : calculé en prenant les variables non tirées. Pour tester la feature j il permutte dans les data les features j et fait 2 risques, si arbre permuté - bon alors feature importante

$$\hat{I}(x^j) = \frac{1}{n_{tree}} \sum_t R_n(h_t, \bar{S}_n^{t,j}) - R_n(h_t, \bar{S}_m^t)$$

VI_ Boosting

A_ AdaBoost as a Greedy Scheme

Idea : on "booste" en ajoutant un poids aux données en fonction de la loss obtenue jusqu'à now.

Weak classifier : classifier with avg training error $\leq 0,5$

Algo : Init : $\omega_1(i) = \frac{1}{m}$ et $H_0 = 0$

Boucle : ① $h_t = \underset{h}{\operatorname{argmin}} \sum_n^m \varepsilon_t(h)$ avec $\varepsilon_t(h) = \mathbb{P}_{i \sim \omega_t}(h(x_i) \neq y_i)$

② $\alpha_t = \frac{1}{2} \log \frac{1-\varepsilon_t}{\varepsilon_t}$ avec $\varepsilon_t = \varepsilon_t(h_t)$

encourage à corriger les trucs mal classifiés ← ③ $\omega_{t+1}(i) = \omega_t(i) \dfrac{e^{-\alpha_t y_i h_t(x_i)}}{Z_{t+1}} = \dfrac{e^{-y_i H_t(x_i)}}{m \prod_n^t Z_s}$

normalisat° tq $\sum \omega_{t+1} = 1$

$Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$

④ $H_t = H_{t-1} + \alpha_t h_t$

Out : $F_T = \operatorname{sign}(H_T)$

Bounding the training error : $R_n(F_T) \leq e^{-2\sum(\frac{1}{2}-\varepsilon_t)^2}$

$R_n(F_T) \leq e^{-2\gamma^2 T}$ où $\gamma \leq (\frac{1}{2} - \varepsilon_t)$

B_ Gradient Boosting

Algo : minimizat° step devient : ① h_t = steepest descent direct° in $\mathcal{H}$

② choose α_t minimize $L(y, H + \alpha h_t)$

easy if finding the best direction is easy

Links

- AdaBoost : $\ell(y,h) = e^{-yh}$
- LogitBoost : $\ell(y,h) = \log(1 + e^{-yh})$
- L_2 Boost : $\ell(y,h) = (y-h)^2$
- L_1 Boost : $\ell(y,h) = |y-h|$
- HubertBoost : $\ell(y,h) = |y-h|^2 \mathbb{1}_{|y-h|<\varepsilon} + (2\varepsilon|y-h| - \varepsilon^2) \mathbb{1}_{|y-h|\geq\varepsilon}$