Apprentissage, réseaux de neurones et modèles graphiques (RCP209)

Méthodes d'agrégation

Marin FERECATU
(prenom.nom@cnam.fr)
http ://cedric.cnam.fr/vertigo/Cours/ml2/

Département Informatique
Conservatoire National des Arts & Métiers, Paris, France

Plan du cours

Objectif

"La raison d'être des statistiques, c'est de vous donner raison." — Abe Burrows

Méthodes d'agrégations :

- Bagging
- Forets Aléatoires
- Boosting

Plan du cours

Avantages et défauts des arbres de décision

Avantages :

- Modèle "white box" : le résultat est facile à conceptualiser et à visualiser
- Ils nécessitent peu de préparation de données (e.g. normalisation, etc.)
- Le cout d'utilisation des arbres est logarithmique
- Capables d'utiliser des données catégorielles et continues
- Capables de gérer des problèmes multi-classe
- Bon comportement par rapport aux outliers
- Gèrent bien les données manquantes

Avantages et défauts des arbres de décision

Problèmes :

- Parfois les arbres générés ne sont pas équilibrés (ce qui implique que le temps de parcours n'est plus logarithmique). Il est donc recommandé d'équilibrer la base de donnée avant la construction, pour éviter qu'il y a une classe dominante (en terme de nombre d'exemples d'apprentissage)
- Sur-apprentissage : parfois les arbres générés sont trop complexes et généralisent mal (solution : élagage, le contrôle de la profondeur de l'arbre et de la taille des feuilles)
- Ils sont **instables** : des changements légères dans les données produisent des arbres très différents. Changements des nœuds proches de la racine affectent beaucoup l'arbre résultant. Ce sont des **estimateurs de variance élevée**.

Estimateurs de variance élevée

Estimateurs de variance élevée :

- Réduction de variance
- Moyenne des estimateurs, calculés sur des données légèrement différentes

Bagging et **Random Forests** : utiliser le hasard pour améliorer les performances des algorithmes de base (arbres de décision CART).

Algorithmes proposés par Breiman, et beaucoup étudiés récemment :

- L. Breiman. *Bagging predictors*, Machine Learning, 24(2), 1996.
- L. Breiman. *Random forests*, Machine Learning, 45, 2001.

Plan du cours

Bagging

Base d'apprentissage :

- Attributs : $A_1, \ldots, A_p$, classe : C
- Données d'apprentissage : $(x_i, y_i), x_i \in R^p, y_i \in R, i = 1, \ldots, N$
- y_i peuvent être des valeurs continues ou discrètes (étiquettes des classes)
- $x_i = (a_1^{(i)}, \ldots, a_p^{(i)})$

On considère $G(x)$ un modèle de prédiction appris sur un échantillon de données $z = \{(x_i, y_i)\}_{i=1}^n$ (e.g. arbre de décision CART)

Bagging

Bagging (Breiman, 1996) :

- On tire au hasard dans la base d'apprentissage B échantillons avec remise $z_i, i = 1, \ldots, B$ (chaque échantillon ayant n points) — appelés échantillons "bootstrap"
- Pour chaque échantillon i on calcule le modèle $G_i(x)$
- Régression : agrégation par la moyenne $G(x) = \frac{1}{B} \sum_{i=1}^{B} G_i(x)$
- Classification : agrégation par vote $G(x) = \text{Vote majoritaire}\,(G_1(x), \ldots, G_B(x))$

Bagging

C'est l'estimateur moyenne qui aide a réduire la variance :

- $X_1, X_2, \ldots, X_n$ variables aléatoires i.i.d. de moyenne μ et variance σ^2
- $\frac{1}{n}(X_1 + X_2 + \cdots + X_n)$ est de variance σ^2/n

Critère performance et calcul de B : l'erreur OOB (Out Of Bag).

- Pour chaque x_k élément de la base d'apprentissage on agrège les erreurs sur les G_i tel que $x_i \notin z_i$ (au lieu de faire un découpage classique test/validation de la base d'apprentissage)
- On choisi B ou l'erreur se stabilise et ne descend plus.

Bagging

Défaut du bagging :

- Les estimateurs G_i ne sont pas en réalité indépendants.
- G_i sont calculés sur des échantillons qui se recouvrent fortement (tirage avec remise), et donc ils sont corrélés.

$X_1, X_2, \ldots, X_B$ variables aléatoires i.d. (mais pas indépendantes) de moyenne μ, variance σ^2 et corrélation $\rho = Corr(X_i, X_j), \forall i \neq j$.
Alors $Y = \frac{1}{B}(X_1 + X_2 + \cdots + X_B)$ est de variance :

$$Var(Y) = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

Quand B est grand le 2eme terme est négligeable mais le 1er non.
L'idée des forets aléatoires est de baisser la corrélation entre les G_i a l'aide d'une étape supplémentaire de randomisation.

Plan du cours

Forêts aléatoires

Forêts aléatoires :

- Amélioration du bagging pour les arbres de décision CART
- Objectif : rendre les arbres utilisés plus indépendants (moins corrélés)
- Bons résultats surtout en grande dimension
- Très simple à mettre en œuvre
- Peu de paramètres

Forêts aléatoires

Forêts aléatoires (Breiman, 2001) :

- On tire au hasard dans la base d'apprentissage B échantillons avec remise z_i , $i = 1, \ldots, B$ (chaque échantillon ayant n points)
- **Pour chaque échantillon i on on tire au hasard q attributs parmi les p existants et on construit l'arbre CART $G_i(x)$ sur ces attributs.**
- Régression : agrégation par la moyenne $G(x) = \frac{1}{B} \sum_{i=1}^{B} G_i(x)$
- Classification : agrégation par vote $G(x) = \text{Vote majoritaire}\,(G_1(x), \ldots, G_B(x))$

Les arbres sont moins corrélés car :

- Ils sont appris sur un ensemble différent des attributs
- Ils sont construits sur des échantillons différents
  En général : $p = \sqrt{p}$

Forêts aléatoires

Forêts aléatoires (Breiman, 2001) :

- On se limite en général à des arbres pas très profonds (pour le Bagging il faut des arbres profonds pour limiter leur corrélation : mais les arbres très profonds souffrent de sur-apprentissage)
- Chaque arbre est petit donc moins performant, mais l'agrégation compense pour ce manquement (chaque attribut se retrouve typiquement dans plusieurs arbres)
- Comme pour le Bagging on utilise l'erreur OOB pour prévenir le sur-apprentissage

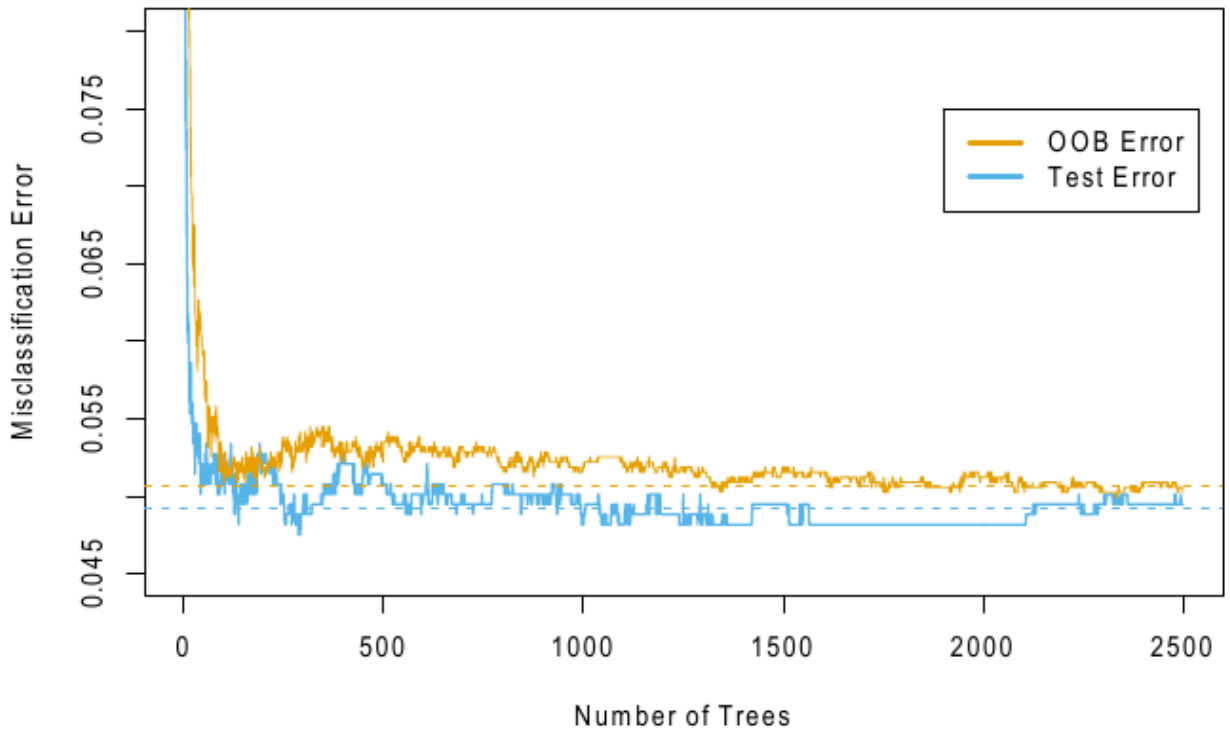
Forêts aléatoires

Paramètres (valeurs par défaut) :

- Classification : $q = \sqrt{p}$, taille nœud minimale 1 ;
- Régression : $q = p/3$, taille nœud minimale 5.

En pratique les valeurs "idéales" dépendent beaucoup de la base (et il faut les trouver par cross-validation.)

Forêts aléatoires



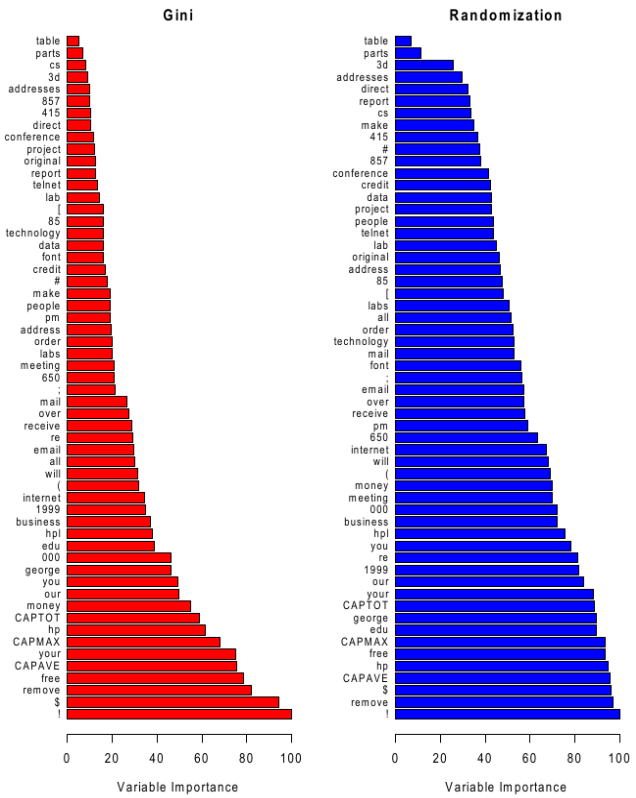
OOB vs erreur de test sur la base "Spambase".

Forêts aléatoires

L'importance des attributs :

- Gini : Le changement dans l'impureté (ou gain d'information) dans chaque noeud cumulé sur tous les arbres de la foret.
- Erreur OOB : tous les échantillons OOB sont évalué par l'arbre et l'erreur mesuré. Ensuite on permute aléatoirement les valeurs sur chaque attribut j et on mesure le taux d'erreur à nouveau. La valeur finale est la dégradation moyenne (changement du taux d'erreurs) sur tous les arbres.

Forêts aléatoires



L'importance des attributs : Gini (gauche) vs OOB error (droite).

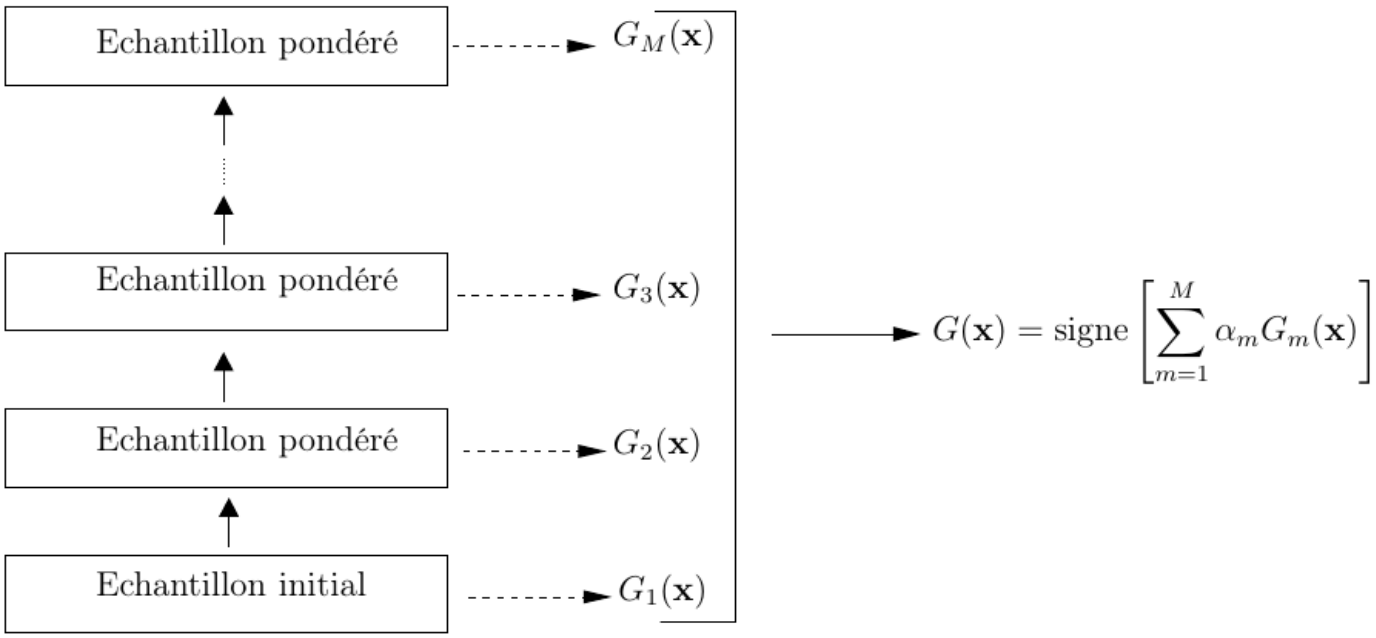
Plan du cours

Boosting

Boosting :

- Combine les sorties de plusieurs classifieurs faibles (weak learners) pour obtenir un résultat plus fort.
- Classifieur faible : un comportement de base meilleur que l'aléatoire (taux d'erreurs sous 0.5 pour une classification binaire)
- Données d'apprentissage : $(x_1, y_1), \ldots, (x_n, y_n)$
- Une famille $\mathcal{G}$ de classifieurs faibles

Boosting



Boosting

Algorithme 1 `AdaBoost`

Entrée :

- $\mathbf{x}$ l'observation à prévoir
- $d_n = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ l'échantillon
- Une réglè faible
- M le nombre d'itérations.

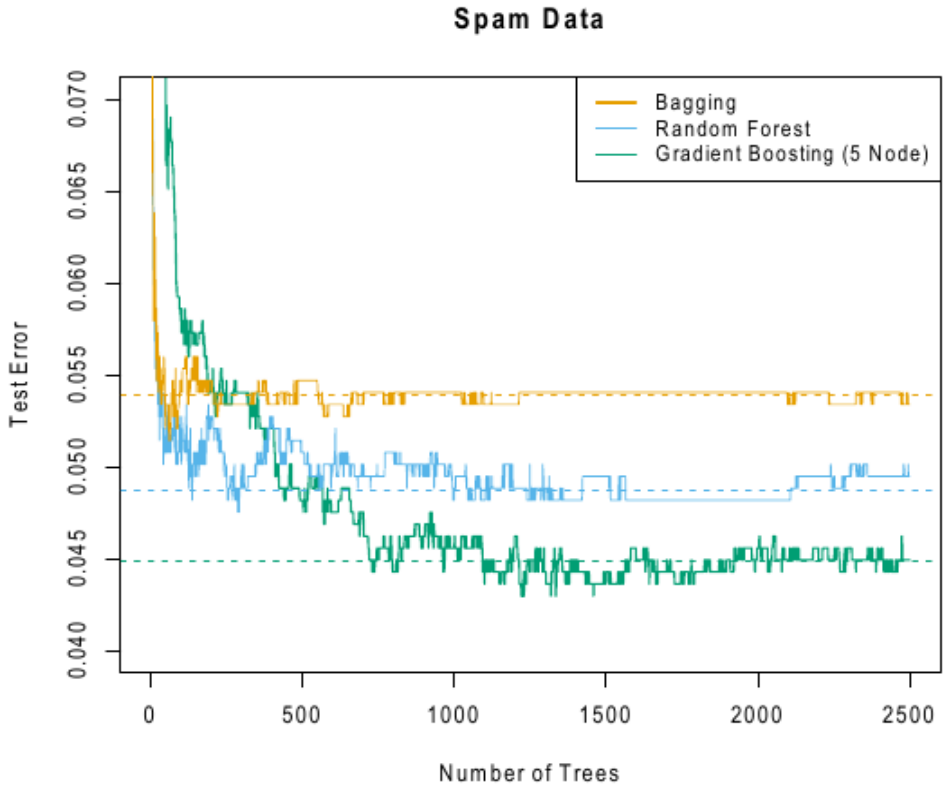
1. Initialiser les poids $w_i = 1/n$, $i = 1, \ldots, n$
2. **Pour** $m = 1$ à M :
   (a) Ajuster la règle faible sur l'échantillon d_n pondéré par les poids $w_1, \ldots, w_n$, on note $g_m(\mathbf{x})$ l'estimateur issu de cet ajustement
   (b) Calculer le taux d'erreur :
   $$e_m = \frac{\sum_{i=1}^{n} w_i \mathbf{1}_{y_i \neq g_m(\mathbf{x}_i)}}{\sum_{i=1}^{n} w_i}.$$
   (c) Calculer : $\alpha_m = \log((1 - e_m)/e_m)$
   (d) Réajuster les poids :
   $$w_i = w_i \exp(\alpha_m \mathbf{1}_{y_i \neq g_m(\mathbf{x}_i)}), \quad i = 1, \ldots, n$$
3. **Sortie** : $\hat{g}_M(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m g_m(\mathbf{x})$.

Boosting

Boosting :

- Étape 2(a) : $g_m(x)$ est le classifieur qui minimise l'erreur pondéré sur la base d'apprentissage $\arg\min_{g_m \in \mathcal{G}} \sum_{i=1}^{n} w_i \mathbf{1}_{y_i \neq g_m(x_i)}$
- L'erreur e_m doit être inférieure à 0.5, sinon α_m devient négative
- L'algorithme minimise l'espérance de la fonction perte "naturelle" $l(y, g(x)) = \mathbf{1}_{y_i \neq g(x_i)}$

Boosting



Références

Livres et articles :

- L. Breiman. *Bagging predictors*, Machine Learning, 24(2), 1996.
- L. Breiman. *Random forests*, Machine Learning, 45, 2001.
- Hastie, Tibshirani, Friedman, *The elements of statistical learning : Data mining, inference, and prediction*, New York, Springer Verlag, 2006
- Laurent Rouvière, *Introduction aux méthodes d'agrégation : boosting, bagging et forêts aléatoires*, polycopié cours, (https ://perso.univ-rennes2.fr/laurent.rouviere)