Ensemble learning

Cours 9

Nicolas Baskiotis
nicolas.baskiotis@lip6.fr

Master 1 DAC

équipe MLIA, Laboratoire d'Informatique de Paris 6 (LIP6)
Université Pierre et Marie Curie (UPMC)

S2 (2015-2016)

Plan

Principe général

- Idée simple : ne pas considérer un mais plusieurs (beaucoup) de classifieurs
- Avantage : réduit la variance si les classifieurs sont indépedants !
- Mais qu'un jeu de données disponible. . .
- Différentes techniques d'échantillonage et d'aggrégation pour varier les classifieurs appris
- Classification : vote majoritaire ou pondéré sur l'ensemble des classifieurs.

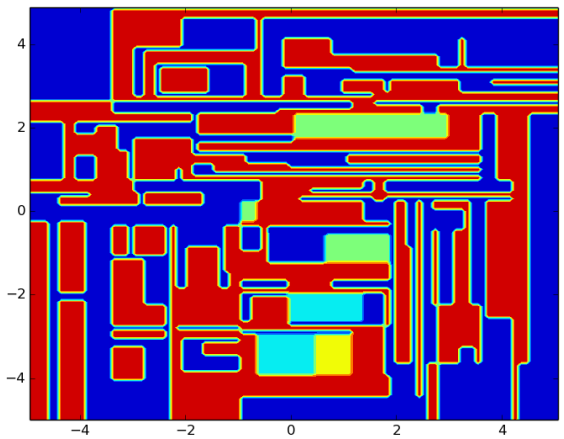
Bagging (Breiman, 1994)

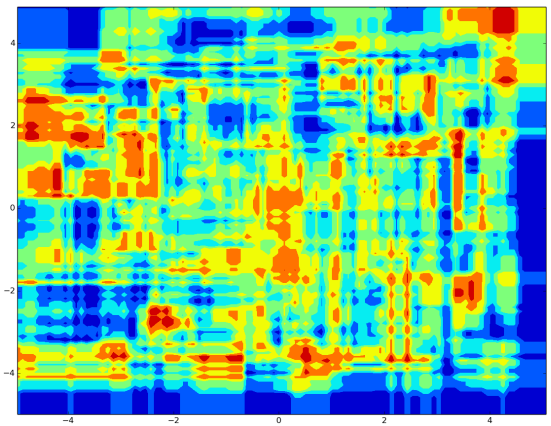
- Bagging : Bootstrap Aggregation : constitution des ensembles par tirage aléatoire **avec remise** d'un ensemble de même taille que l'original.
- Chaque donnée a une probabilité de $(1 - 1/n)^n$ d'être choisi comme donnée test
- Les données d'apprentissage vont contenir $1 - (1 - 1/n)^n\% = 63.2\%$ des instances initiales.

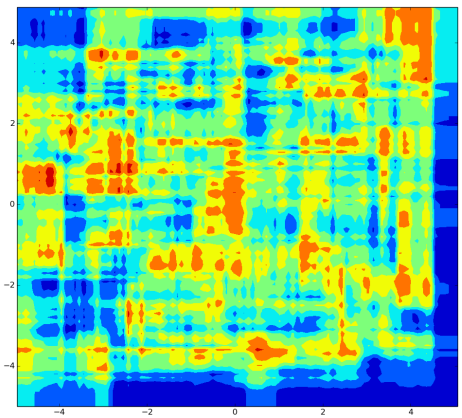
Un exemple : plusieurs arbres = une fôret

Principe

- A l'origine pour des considérations computationelles
- Deux facteurs d'aléa :
  - chaque arbre est appris sur un ensemble bootstrap de l'initial (bagging)
  - à chaque nœud, un sous-ensemble des dimensions est considéré uniquement, tiré aléatoirement.
- Décision au vote majoritaire (ou en moyenne pour les arbres de régressions).
- Remarques : Effet de la profondeur ? Sur-apprentissage ?







Boosting

Principe

- Pourquoi faire de l'aléatoire quand on connait où on fait l'erreur ?
- Trouver plein de petits classifieurs approximativement bons sur de petites régions peut être beaucoup plus facile que de trouver un classifieur complexe sur tout l'espace.
- Idée : aggréger plein de petits classifieurs, appris séquentiellement, chacun corrigeant les erreurs des précédents.

$\Rightarrow$ Besoin de classifieurs faibles ! (pourquoi ?)

Questions

- Qu'est ce qu'un classifieur faible ?
- Comment modifier la distribution à chaque pas pour prendre en compte les erreurs ?
- Comment combiner les classifieurs ?
- Combien de classifieurs apprendre ?

Boosting

Principe

- Pourquoi faire de l'aléatoire quand on connait où on fait l'erreur ?
- Trouver plein de petits classifieurs approximativement bons sur de petites régions peut être beaucoup plus facile que de trouver un classifieur complexe sur tout l'espace.
- Idée : aggréger plein de petits classifieurs, appris séquentiellement, chacun corrigeant les erreurs des précédents.

$\Rightarrow$ Besoin de classifieurs faibles ! (pourquoi ?)

Questions

- Qu'est ce qu'un classifieur faible ?

$\Rightarrow$ classifieur peu expressif, arbres de faibles profondeurs, perceptrons . . .

- Comment modifier la distribution à chaque pas pour prendre en compte les erreurs ?

$\Rightarrow$ Considérer une distribution D_t différente à chaque pas de temps

- Comment combiner les classifieurs ?

$\Rightarrow$ Somme pondérée des classifieurs

AdaBoost

Principe

- $D = \{x^i, y^i\}$ un ensemble de données, distribution $D_t(i) = w_t^i$ sur ces données au temps t
- $h = \{h_t : h_1, \cdots, h_T\}$ un ensemble de classifieurs,
- $\alpha = \{\alpha_t : \alpha_1, \cdots, \alpha_T\}$ un ensemble de réels,
- $f_T(x) = \sum_{i=1}^{T} \alpha_t h_t(x) =< \alpha, h >$, $F_T(x) = sign(f_t(x))$ le classifieur pondéré.
- Objectif : trouver $(h^\star, \alpha^\star) = argmin_{h,\alpha} \frac{1}{N} \sum_i 1_{F(x^i) \neq y^i}$

Algorithme

1. Initialiser la distribution : $D_0(i) = \frac{1}{D}$
2. Apprendre h_t sur D_t
3. Calculer l'erreur $\epsilon_t = \sum_i D_t(i) 1_{h_t(x^i) \neq y^i}$
4. Fixer $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$
5. Fixer $D_{t+1}(i) = \frac{1}{Z_t} D_t(i) e^{-\alpha_t y_i h_t(x_i)}$

Remarques

Considérations sur les poids

- $\epsilon_t < \frac{1}{2} \Rightarrow \alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t}) > 0$
- $\epsilon(h_a) < \epsilon(h_b) \Rightarrow \alpha_a > \alpha_b$
- $e^{-y\alpha_t h_t(x)} = \begin{cases} e^{-\alpha_t} < 1 & \text{si } h_t(x) = y \\ e^{\alpha_t} > 1 & \text{si } h_t(x) \neq y \end{cases}$

Considérations sur la distribution

- $D_{t+1}(i) = \frac{1}{Z_t} D_t(i) e^{-\alpha_t y^i h_t(x^i)} = \frac{1}{Z_t Z_{t-1}} D_{t-1}(i) e^{-y^i(\alpha_t h_t(x^i) + \alpha_{t-1} h_{t-1}(x^i))}$
  $\cdots = \frac{1}{Z_t \cdots Z_1} D_1(i) e^{-y^i(\alpha_t h_t(x^i) + \cdots + \alpha_1 h_1(x^i))}$
- On montre que $Z = Z_1 \cdots Z_t = \frac{1}{N} \sum_{i=1}^{N} e^{-y^i f_t(x^i)}$
- Et que $Err(F) \leq Z$

Remarques

Sur le bagging

- Très utilisé ! (kinect, les gagnants de netflix)
- Facile à mettre en place, peut traiter de grosses masses de données (parallélisation), en apprentissage et en inférence

Boosting

- Classifieurs faibles : Stump (arbre à un niveau), naive bayes, perceptron,. . .
- Adaptable sous beaucoup d'autres formes (gradient tree boosting, gradient boosting)
- Adapté au très grande masse de données et données sparse (ciblage publicitaire par exemple)