Outils avancés du data scientist

Data science
Master 2 ISIDIS

Sébastien Verel
verel@lisic.univ-littoral.fr
http://www-lisic.univ-littoral.fr/~verel

Université du Littoral Côte d'Opale
Laboratoire LISIC
Equipe OSMOSE

Plan

Quelques mots d'introduction

- Algorithmes d'apprentissage très performant
- "Rapide" à entrainer, parallélisable, implémenté dans beaucoup de framework (R, sklean)
- Moins facile d'interprétation que les arbres de décision, mais meilleure précision
- Méthode ensembliste (à ensemble)

Breiman, Leo (2001). "Random Forests". Machine Learning 45 (1) : 5-32.

Idée de base

- Méthode ensembliste :
  utilise un ensemble de prédicteurs "simples"
- Prédicteur simple :
  arbre de décision "simple"
- Prédiction :
  - Vote de majorité dans le cas clustering
  - Moyenne des prédictions dans le cas régression
- Clé de la phase d'apprentissage :
  construire un ensemble d'arbres variés

Principe

random forest = tree bagging + features sampling

Les arbres de décision dépendent trop de l'échantillon.

Pour construire chaque arbre, vision parcellaire du problème :

- Tirage (avec remise) d'un sous-ensemble d'exemples
  $\Rightarrow$ bagging
- Tirage (avec remise) d'un sous-ensemble de variables
  $\Rightarrow$ features sampling

Réduction de la variance par la forêt

$$V_{forest} = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

où :

- σ^2 : variance d'un arbre de décision
- B : nombre d'arbres
- ρ corrélation entre les arbres

La feature sélection peut faire baisser le coefficient ρ
(par défaut $\sqrt{n}$ features sont sélectionnées)

Variantes

- Extremely randomized trees :
  Pour chaque variable sélectionnée, le split lui-même est aussi choisi aléatoirement
- rotation forest :
  Analyse en composante principale (ACP) avant de construire l'arbre

Geurts, Pierre and Ernst, Damien and Wehenkel, Louis, Extremely randomized trees, Machine Learning, vol. 6, 1, pp. 3-42, 2006.

Rodriguez JJ1, Kuncheva LI, Alonso CJ, Rotation forest : A new classifier ensemble method, IEEE Trans Pattern Anal Mach Intell. 2006 Oct ;28(10) :1619-30.

Boosting

Bibliographie

- adaBoost pour adaptive boosting (prix Gödel 2003) :

  Yoav Freund et Robert Schapire, A desicion-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, vol. 55, Num. 1, ?1997, p. 119-139

- Gradient boosting :

  J.H. Friedman, Greedy function approximation a gradient boosting machine, Ann. Statist., Vol., Num. 5 (2001), 1189-1232.

Principe du boosting

Principe

- Construction itérative par un meta-algorithme des arbres (plus généralement un weak learner)
- Les erreurs de l'itération précédente sont prises en compte pour la construction suivante

  Les weak leaners sont "boostés"

Boosting vs. random forest

- Random forest :
  Vote à la majorité
- Boosting :
  Somme pondérée des weak learners

$$H(x) = \text{sign}(\sum_{i=1}^{B} \alpha_i h_i(x))$$

L'algorithme Adaboost

Initialiser la distribution des exemples $\forall i = 1, \ldots, m,\ D_1(i) = \frac{1}{m}$

for $t = 1, \ldots, T$ **do**

$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \epsilon_t$ avec ϵ_t l'erreur de classification pondérée :

$$\epsilon_t = \sum_{i=1}^{m} D_t(i).[y_i \neq h(x_i)]$$

Poids du classifieur : $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$

mise à jour des pondérations :

$$\forall i = 1, \ldots, m,\ D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

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

end for

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Gradient boosting : Principe

gradient boosting = descente de gradient + boosting

La suite sur les slides :
`http://orbi.ulg.ac.be/bitstream/2268/163521/1/slides.pdf`