Data Mining 2 : Construction d'Ensembles de Classifieurs

TP2 - Le Boosting

2014 - 2015

Rappels

On rappelle ci-dessous le principe générique du boosting pour la classification à deux classes :

1. On dispose d'une base d'exemples D : $\{((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n))\}$
2. On suppose $y_i \in \{-1, 1\}$
3. Pour $t = 1, \ldots, T$
   (a) Construire une 'distribution' $D^{(t)}$ sur $1, \ldots, n$
   (b) Trouver un classifieur faible :
   $$h_t : X \to \{-1, 1\}$$
   tel que ϵ_t soit petit :
   $$\epsilon_t = P_{D^{(t)}}(h_t(\mathbf{x}_i) \neq y_i)$$
4. Combiner les h_t pour produire le classifieur final h_c

L'algorithme de référence qui implémente ce principe est appelé Adaboost. Il a initialement été introduit pour la classification à deux classes et étendu ensuite à la classification multiclasses, dans de multiples versions. Nous donnons en algorithme 1, l'algorithme d'apprentissage d'Adaboost pour la classification à 2 classes et en algorithme 2 une version multiclasses, appelée Adaboost.M1.

Algorithm 1 AdaBoost (2 classes)

ENTRÉES: : D, l'ensemble d'apprentissage de n données, T, un nombre d'itérations et $\mathcal{L}$, un apprenant faible
SORTIES: : H, l'EoC, et Θ, des poids associés à chaque h_t

$D^{(1)} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$
pour $t = 1, ..., T$ **faire**
  $h_t = \mathcal{L}(D^{(t)})$
  $\widehat{\epsilon_t} = \sum_i \mathbb{1}(h_t(x_i) \neq y_i) D_i^{(t)}$
  $\theta_t = \frac{1}{2} ln(\frac{1-\widehat{\epsilon_t}}{\widehat{\epsilon_t}})$
  pour $i = 1, ..., n$ **faire**
    $D_i^{(t+1)} = \frac{D_i^{(t)}}{Z_t}\, exp(-\theta_t y_i h_t(x_i))$
  fin pour
  $H = H \cup h_t$ et $\Theta = \Theta \cup \theta_t$
fin pour

1 Exercices

1.1 Adaboost pour la classification à 2 classes

Écrivez deux fonctions matlab : `adaboostLearn` et `adaboostPred`, implémentant respectivement l'apprentissage et la prédiction de la méthode Adaboost avec des arbres de décisions. Pour cela vous pouvez utiliser la toolbox PRTools :

- `treec` : qui permet de faire l'apprentissage d'un arbre de décision

Algorithm 2 AdaBoost.M1 (multiclasses)

ENTRÉES: : D, l'ensemble d'apprentissage de n données, T, un nombre d'itérations et $\mathcal{L}$, un apprenant faible
SORTIES: : H, l'EoC, et Θ, des poids associés à chaque h_t

$D^{(1)} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$
pour $t = 1, ..., T$ **faire**
  $h_t = \mathcal{L}(D^{(t)})$
  $\widehat{\epsilon_t} = \sum_i \mathbb{1}(h_t(x_i) \neq y_i) D_i^{(t)}$
  $\theta_t = ln(\frac{(C-1)(1-\widehat{\epsilon_t})}{\widehat{\epsilon_t}})$
  pour $i = 1, ..., n$ **faire**
    $D_i^{(t+1)} = \frac{D_i^{(t)}}{Z_t}\ exp(\theta_t \mathbb{1}(h_t(\mathbf{x}_i) \neq y_i))$
  fin pour
  $H = H \cup h_t$ et $\Theta = \Theta \cup \theta_t$
fin pour

- `stumpc` : qui permet de faire l'apprentissage d'une souche binaire de décision
- `gendatw` : qui permet de ré-échantillonner un ensemble de données en fonction d'un vecteur de poids fournit en argument.
- la prédiction d'un classifieur pour une donnée ou un ensemble de données est obtenue par : `P = Ds*w`, où Ds est une donnée ou un ensemble de données à prédire, w est un classifieur appris à l'aide d'une fonction de la PRTools et P une structure permettant d'accéder aux prédictions (`P.nlab`) ou aux scores d'appartenance aux classes (`P.data`). Vous pouvez utiliser la commande `struct(P)` pour en visualiser les champs disponibles.

Testez vos fonctions à l'aide des bases de données fournies sur la plateforme Moodle (*datasets.zip*). Cette archive contient :

- 2 bases de données à 2 classes : *Ionosphere* et *Diabetes*
- 3 bases de données à 7, 4 et 8 classes, respectivement. Les bases nommées *synthX* sont des bases de données synthétiques à 2 dimensions que l'on peut visualiser avec la fonction `scatterd` de la PRTools.

1.2 Adaboost pour la classification à C classes ($C > 2$)

Implémentez maintenant la version multiclasses d'Adaboost (apprentissage et prédiction), appelée Adaboost.M1. Vous pouvez utiliser des souches binaires de décision ou des arbres de décisions. Vous aurez également besoin d'accéder aux scores d'appartenance à chaque classe, fournis pour chaque donnée à prédire : `P.data` (*cf. explications de la sous-section précédente*).

Testez vos fonctions sur les 3 bases de données multiclasses fournies dans l'archive *datasets.zip*. Vous pouvez visualiser les résultats obtenus sur les deux bases synthétiques à l'aide des fonctions `scatterd` et `plotc`, comme illustré dans le fichier *demo.m* fournit sur la plateforme Moodle.

Question bonus : implémentez Adaboost.M2, dont vous avez vus l'algorithme d'apprentissage en cours.