Data Mining 2 : Construction d'Ensembles de Classifieurs

TP1 - Le Bagging

2014 - 2015

Rappels

- On rappelle ci-dessous le principe de la méthode de bootstrap :
  - On dispose d'un échantillon de données $D = (x_1, x_2, x_3, ..., x_n)$, représentatives d'une population de données, et on s'intéresse à la statistique $s(D)$.
  - On forme K nouveaux échantillons bootstrap $D^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, ..., x_p^{(k)})$, générés par tirages aléatoires (avec remise) de p données dans D.
  - On calcule $s(D^{(k)})$ pour chaque échantillon bootstrap.
  - On peut ensuite calculer :

$$\widehat{s} = \frac{1}{K} \sum_{k=1}^{K} s(D^{(k)})$$

- On rencontre beaucoup d'applications intéressantes de cette méthode en Statistiques comme en Data Mining. Par exemple le bootstrap :
  - permet de calculer simplement l'erreur-standard et le biais d'un estimateur.
  - permet également d'en calculer la variance.
  - permet tout particulièrement de calculer des intervalles de confiance sur des grandeurs statistiques de plusieurs façons différentes (méthodes de l'erreur-standard, des pourcentiles simples, des pourcentiles corrigés pour le biais, etc.).
  - peut se substituer à des méthodes d'inférences statistiques "classiques" lorsque que certaines conditions nécessaires ne sont pas remplies (ex : test *t de student* d'égalité des moyennes, pour lequel les conditions sont parfois "trop fortes")
  - etc.

- Une de ses applications nous intéresse tout particulièrement : la génération d'ensembles de classifiers (Bagging). C'est la méthode que vous allez tester dans ce TP.

1 Exercices

1.1 Générer un ensemble bootstrap

Écrire une fonction `drawBootstrap` qui prend en entrée deux paramètres :

1. le nombre d'individus dans la population initiale
2. le nombre d'individus à tirer

et qui retourne 2 listes d'indices :

1. `bag` qui contient les indices des données tirées aléatoirement
2. `oob` qui contient les indices des données de l'*out-of-bag*.

Ainsi, si `D` est une matrice qui contient les données initiales (une donnée par ligne), `D[bag,:]` représente l'ensemble bootstrap généré. `D[oob,:]` représente quant à lui les données n'apparaissant pas dans l'ensemble bootstrap.

1.2 Estimer un moment statistique

Vous trouverez sur la plateforme *moodle* un fichier contenant des réalisations d'une variable aléatoire réelle qui suit une loi asymétrique. On souhaite estimer la valeur du coefficient de dissymétrie (*skewness*) à l'aide de la méthode de bootstrap. Pour rappel, ce coefficient est le moment d'ordre trois de la variable centrée réduite :

$$\gamma = \mathbb{E}\left[\left(\frac{X-\mu}{\sigma}\right)^3\right]$$

Écrivez une fonction `skewness` qui permet d'estimer la valeur du coéfficient de dissymétrie en utilisant la méthode de bootstrap. Cette fonction fournira en sortie une estimation de ce coefficient ainsi que l'erreur standard associée.

1.3 Bagging

Vous devez maintenant implementer la méthode de Bagging.

Pour cela, vous pouvez utiliser la librairie matlab de reconnaissance de formes *PRTools* [1]. Cette librairie implémente un grand nombre de classifieurs, dont deux classifieurs que vous allez utiliser pour tester votre fonction de Bagging. Pour pouvoir utiliser la librairie il faut :

1. télécharger l'archive contenant la librairie depuis la plateforme moodle
2. créer un répertoire PRTools dans votre répertoire de travail pour y décompresser le contenu de l'archive.
3. ajouter au *path* matlab le chemin de ce répertoire contenant l'ensemble des fichiers de la librairie.

Vous avez maintenant à votre disposition les fonctions :
- `knnc` : qui implémente un algorithme de K-Plus Proches Voisins.
- `treec` : qui implémente un algorithme d'apprentissage d'Arbres de Décisions.

Ces fonctions vous permettront de générer les classifieurs composants votre ensemble de classifieurs 'baggés'.

Un script matlab de demonstration (*demo.m*) vous est fourni via *moodle*, qui donne un exemple d'utilisation de ces deux fonctions. Vous êtes bien sûr invité à consulter l'aide ainsi que le manuel d'utilisation de la librairie pour plus d'informations.

On met également à votre disposition sur *moodle* 5 bases de données [2] qui vous permettront de tester votre code. Il faudra pour cela diviser chacune de ces bases en deux sous-ensembles : un sous-ensemble d'apprentissage et un sous-ensemble de test.

Question bonus : modifiez votre fonction pour que celle-ci fournisse également une estimation de l'erreur en généralisation en utilisant les données out-of-bag.

1. `http://www.37steps.com/software/`
2. bases de données publiques issues de l'*UCI Machine Learning repository* `http://archive.ics.uci.edu/ml/index.html`