Machine Learning
TD - Fondamentaux en Apprentissage

M2 Informatique - BI&A

Guillaume Metzler

Institut de Communication (ICOM)
Université de Lyon, Université Lumière Lyon 2
Laboratoire ERIC UR 3083, Lyon, France

guillaume.metzler@univ-lyon2.fr

Abstract

Ce TD a pour objectif de déterminer vos connaissance en Machine Learning, notamment sur le plan pratique. Est-ce que le processus d'apprentissage d'un modèle est connu/maîtrisé mais aussi de voir vos différents réflexes face à des jeux de données qui peuvent présenter différentes caractéristiques.

Etude des jeux de données

Vous pourrez trouver plusieurs jeux de données à l'adresse suivante

Téléchargement des jeux de données.

Ils pourront vous servir tout au long de cette première séance. Vous pourrez également trouver un code Python permettant de mettre en forme les différents jeux de données afin que ces derniers soient prêts à être employés

Préparation des données

On pourra effectuer les différentes étapes :

- regarder le code précédent afin d'identifier les différentes étapes
- détecter d'éventuelles valeurs manquantes et faire de l'imputation
- identifier les caractéristiques des jeux de données (taille, dimension, type de problème, ratio des classes)

Expériences

Une fois que les données sont prêtes, vous essaierez de mettre en oeuvre le processus d'apprentissage à l'aide de vos connaissances actuelles dans le domaine.

- processus d'apprentissage
- choix d'une mesure de performance appropriée
- mise en oeuvre de méthodes de pré-apprentissage sur les données
- mise en oeuvre de plusieurs algorithmes
- comparaison des performances des algorithmes (performances mais aussi temps d'apprentissage)
- ...

A la fin de cette partie, vous devez avoir avoir un tableau qui synthétise les résultats obtenus à l'aide des différentes méthodes sur les jeux de données employés pour ce premier travail.

Quelques librairies ou fonctions utiles

```
# Pour créer les ensembles d'entraînement et de test
from sklearn.model_selection import train_test_split

# Pour créer les différents folds pour la k-CV
from sklearn.model_selection import StratifiedKFold

# Pour normaliser les données
from sklearn.preprocessing import Normalizer

# Matrice de confusion pour calculer différentes métriques.
from sklearn.metrics import confusion_matrix

# Quelques algos comme le k-NN et le SVM
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
```

On donnera ci-dessous une façon d'apprendre un classifieur, par exemple un SVM, sous python. On supposera que les données se trouvent dans des objets $X_{\text{train}}, Y_{\text{train}}$ et X_{test}. Cet algorithme dépend d'un hyper-paramètre C dont il va falloir déterminer, par cross-validation, la valeur optimale.

```
### Apprentissage d'un SVM linéaire ###

# Instanciation du modèle
clf = svm.SVM(C = ..., kernel = "linear")

# Apprentissage des paramètres du modèle
clf.fit(X_train,Y_train)

# Prédictions
pred = clf.predict(X_test)
```