Rapport TME1: Arbres de décision, sélection de modèle

Yuhan WANG et Tianyu WANG

June 2020

Ce rapport comprend les réponses du questions posées dans l'énoncé et les figures. Les codes éxécutables sont dans tme1.ipynb

1 Introduction

Un arbre de décision est un outil d'aide à résoudre le problème de classification qui représente un ensemble de choix sous la forme graphique d'un arbre. L'exemple comportant n symboles à classifier est sous la forme $x = (x_1, x_2, ..., x_d) \in \mathcal{R}^d$. $\mathcal{Y} = (y_1, y_2, ..., y_m)$ correspond aux labels pour le training data. L'apprentissage par arbre de décision désigne une méthode basée sur l'utilisation d'un arbre de décision comme modèle prédictif. C'est une technique d'apprentissage supervisé. L'apprentissage s'effectue de manière récuisive. La racine de l'arbre répresente l'ensemble de tous les samples, à chaque noeud on effectue un test sur une dimension et chaque feuille represente le résultat de classificatioin.

2 Définitions

2.1 Entropie

$x = (x_1, x_2, ..., x_d)$ comportant d symboles, chaque symbole x_i ayant une probabilité p_i d'apparaître dans l'ensemble d'apprentissage. L'entropie **H** pour un ensemble des exemples peut etre calculé:

$$H(X) = -\sum_{yinY} p_y * log(p_y)$$

2.2 Entropie conditionnelle

L'entropie conditionnelle permet de calculer l'homogénéité entre les objets dans une partition. Avec un split de n-aire $P = (P_1, P_2, ..., P_n)$, l'entropie condi-

tionnnelle à P s'écrit:

$$H(X|P) = -\sum_{i=1}^{n} p(y|P_i) * H(X|P_i)$$

2.3 Gain information

Le gain d'information est un critere important pour le choix sur quelle dimension faire le test. Plus cette valeur est grande, plus cette dimension est importante.

gain information=entropie-entropieconditionnelle

Expériences préliminaires

---

1 Base imdb

Cette base de donnée contient 4587 exemples des films. Chaque exemple est composé de 32 critères. Le vecteur datay correspond aux labels des exemples: datay=+1 ou -1.

1.3 Calculer pour chaque attribut binaire l'entropie et entropie conditionnelle, calculer également leur différence. Quelle est la meilleur attribut pour la première partition?

entropy(datay)=0.9869
Pour chaque critère, on teste d'abord s'il est un attribut binaire, si oui on calcule le gain information après la division par cet attribut. La différence entre l'entropie et l'entropie conditionnelle, donc le gain information est le suivant: Une valeur 0 pour le gain information veut dire la purité des informations ne change pas après cette partition, elle est donc inutile. Le meilleur attribut pour la première partition est attribut *drama*.

1.4 Apprendre quelques arbres de profondeurs différentes

1.5 Comment les scores évoluent-ils en fonction de profondeur?

On prend une list max_depth=[5,10,15,20,25]. En testant sur la base d' apprentissage, le score augemente avec l'agmentatioin du profondeur.

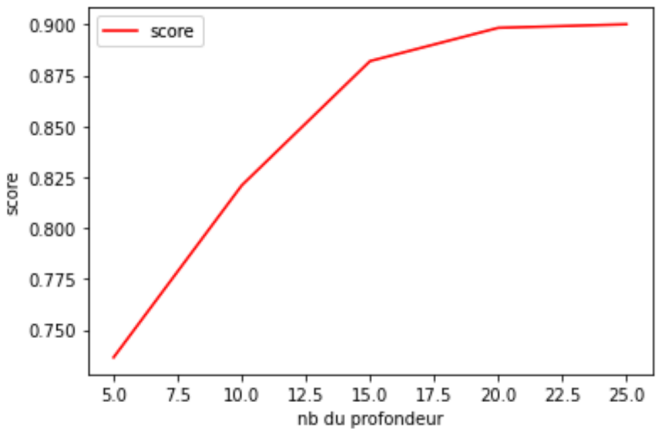

Figure 1: Courbe score

1.6 Les scores sont-ils un indicateur fiable du comportement de l'algorithme?

Non. Il faut le tester avec un ensemble d'exemples non conclus dans la base d'apprentissage: diviser les data vers training set et test set.

Sur et sous apprentissage

Pour réaliser la partition, on a fourni dans decisiontree.py une fonction *splitBase(prc,datax,datay)* où prc représente le pourcentage pour le training set.

1.7 Tracer les courbes de l'erreur en apprentissage et de l'erreur en fonction de différents partitionnement et de la profondeur du modèle

1.8 Que remarquez vous quand il y a peu d'exemple à apprendre? De même quand il y en a trop?

Quand il y a peu ou trop d'exemple d'apprentissage, l'arbre de décision n'a pas de bonne performance sur le test set. Mais quand il y en a beaucoup, sa performance pour le training set est la meilleure.

1.9 Comment améliorer?

Il faut réfléchir le façon d'echantillonage. Ex: stratified sampling ou bootstrap sampling.

```
Out[226]: <function matplotlib.pyplot.show(*args, **kw)>
```

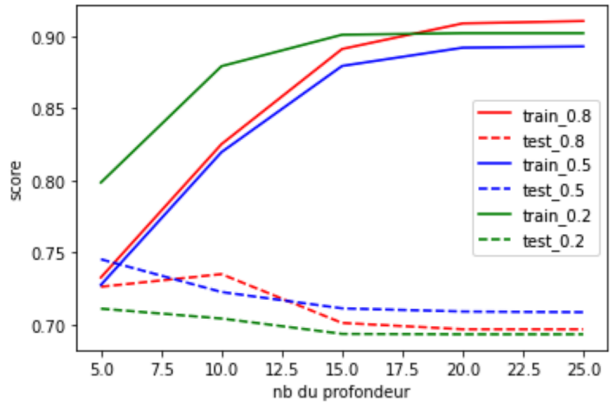

Figure 2: Courbe score

Validation croisée

Pour réaliser la validation croisée, on a fourni dans decisiontree.py une fonction *validatioin_croise(datax,datay,n)* où n représente le nombre de partition.

```
<function matplotlib.pyplot.show(*args, **kw)>
```

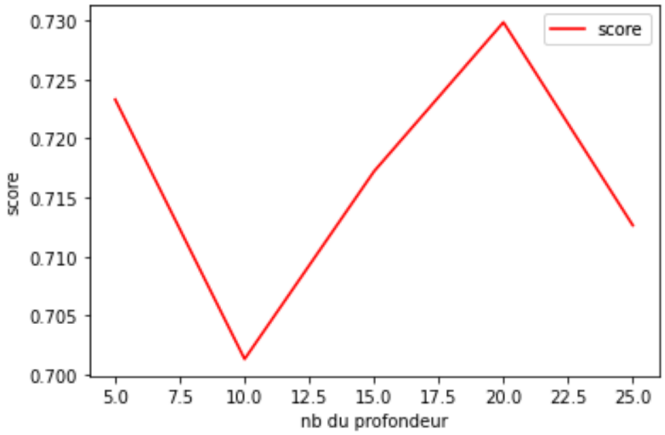

Figure 3: Score en utilisant la validation croisée