Sorbonne université

Rapport de ARF

TME 1 - 6

Samu Tamminen

Année 2017-2018

Table des matières

1 TME 1 : Arbres de décision, sélection de modèles

1.1 Q1.3 Interpretation de l'entropie

Calculer également la différence entre l'entropie et l'entropie conditionnelle pour chaque attribut. A quoi correspond une valeur de 0 ? une valeur de 1 ? Quel est le meilleur attribut pour la première partition ?
Quand la différence entre l'entropie et l'entropie conditionnelle est 0 quand la partition n'est pas du tout bon et un, quand la partition est le meilleur possible.

1.2 Q1.4 - 6 La profondeur d'arbre

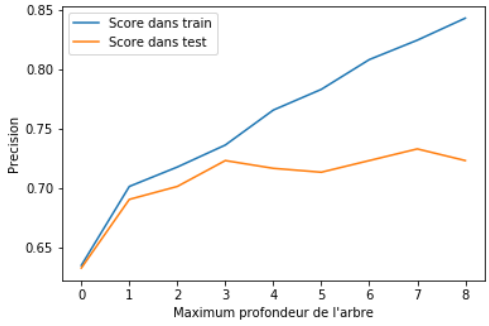
En augmentant la profondeur maximale d'arbre, on obtient meilleur score. Mais ces scores ne sont pas un indicateur fiable, si nous ne faisons pas différence entre les données training et test.

1.3 Q1.7 - Sur et sous apprentissage

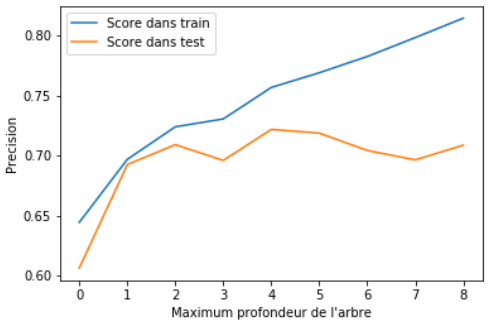
Le partition 0.8 / 0.2 dans Figure 1 (c) semble d'etre le plus robust pour test et training. La difference difference entre le score dans aprentissage et test est le plus petit parmis (a),(b) et (c).

Quand il y a que peu d'examples d'apprentissage, l'erreur est plus éleve dans le test. Par contre quand on a beacoup des d'examples d'aprentissage, l'erreur dans le test est moins d'élevé.

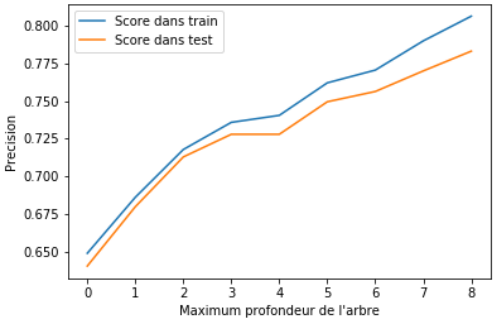
FIGURE 1 – Score d'arbre de decision en faisant varier le profondeur



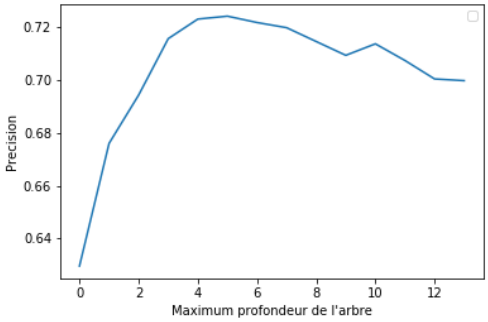
(a) Donnée aprentissage et test : 0.2 / 0.8



(b) Donnée aprentissage et test : 0.5 / 0.5



(c) Donnée aprentissage et test : 0.8 / 0.2



(d) Validation croisee

Avec validation croisée on obtient un score plus fiable et stable. Dans la figure Figure 1 (d) nous pouvons constater, comment le score augmente au début avec la profondeur maximale, mais à la fin il tombe. Ça c'est à cause de sous et sur apprentissage. Au début, on généralise beaucoup avec un arbre peu profond (sous apprentissage). Quand la profondeur se monte sur cinq, nous commençons à apprendre les données d'apprentissage à la façon trop détaillée et le score sur le test se tombe (sur aprentissage). La profondeur maximale 4 semble optimal.

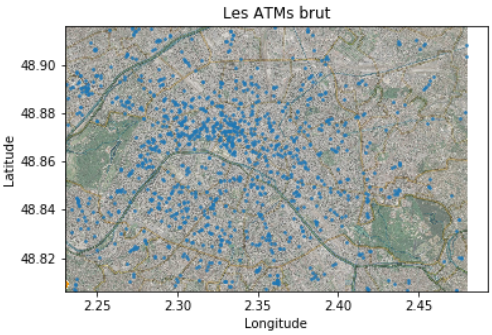
2 TME 2 : Estimation de densité

Nous sommes intéressés de densité des services différents de Paris.

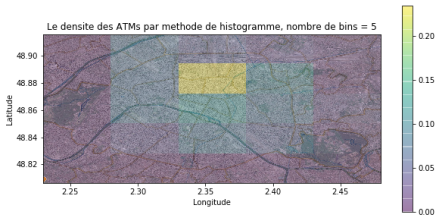
2.1 Méthode des histogrammes

La méthode la plus simple, méthode des histogrammes, il s'agit de partager la valeur continue (latitude et longitude) à les intervalles fixés et compter, combien d'occurrences chacun contient.

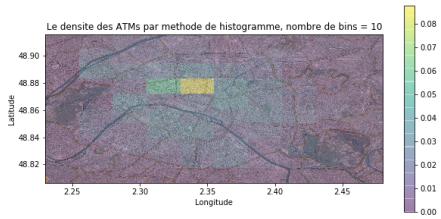
FIGURE 2 – Densite par méthode des histogrammes



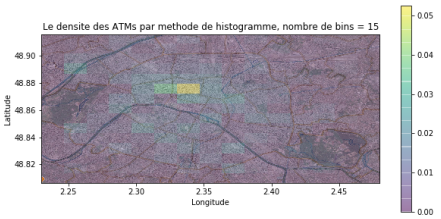
(a) Les ATMs à Paris



(b) Histogramme avec 5 intervalles



(c) Histogramme avec 10 intervalles



(d) Histogramme avec 15 intervalles

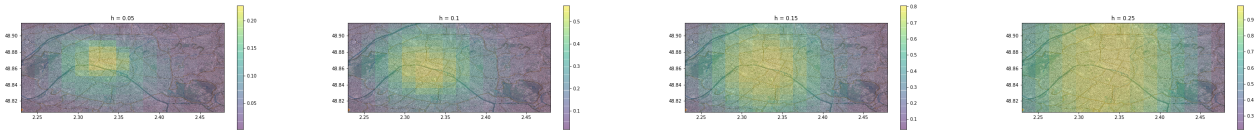
On peut voir bien avec chaque discretization de Figure 2, que la densité maximum des ATMs est dans 9iéme arrondissement. Par contre, la densite n'est pas très visible dans les endroits où la densité est moins forte (par exemple 13iéme arrondissement). Surtout avec 15 étapes, nous ne voyons pas très bien.

2.2 Méthode à noyaux

Comme la méthode de histogramme ne permet pas de visualiser les atms bien avec 15 étapes, je vais fixer le variable étapes à 15 pour l'estimation par noyau. L'estimation par noyau a une variable

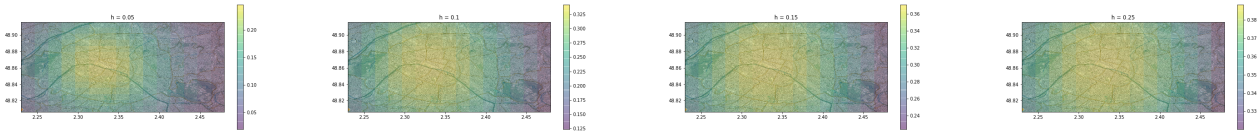
h qui est taille de lissage. On va varier d'abord h.

FIGURE 3 – Parzen comme fonction de Noyau



Dans Figure 3, si h >= 0.25, l'estimation est "oversmoothed". h = 0.15 l'air le plus optimal, car il prend en compte les points partout à Paris, mais la densité est également visible.

FIGURE 4 – Gauss comme fonction de Noyau



Avec le gaussien (Figure 4), le meilleur paramètre de lissage est peut-être 0.05.

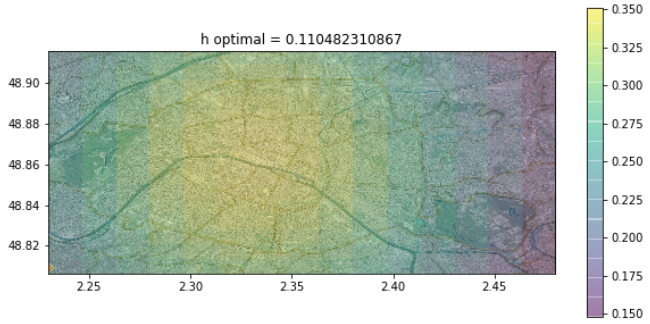
2.2.1 Comment choisir de manière automatique le meilleur taille de lissage h ?

Selon [A rule-of-thumb bandwidth estimator, 2018], nous pouvons estimer l'optimal taille de lissage h pour noyau de Gauss. L'estimation est definit dans Équation 1.

$$h = \left(\frac{4\hat{\sigma}^5}{3n}\right)^{\frac{1}{5}} \approx 1.06\hat{\sigma}n^{-1/5} \tag{1}$$

Nous obtenons h=0.110482310867 pour les données des ATMs à Paris, visualisé dans Figure 5.

FIGURE 5 – Densité avec la méthode de noyaux (Gauss) et lissage optimale



L'optimal "rule-of-thumb" estimateur minimise l'erreur de moindre carrée intégré (Mean integrated squared error). Il permet de montrer le "vraie" distribution, que les POIs vont suivre. J'ai visualisé également les autres POIs avec le noyaux de Gauss et l'optimal h. J'ai constaté que les visualisations ne sont pas très différentes. Les différentes POIs sont distribués à Paris plus ou moins à la même façon : en centre-ville la densité est grande, d'ailleurs moins. Donc je pense que ça sera difficile de classifier les POIs seulement par rapport ses localisations.

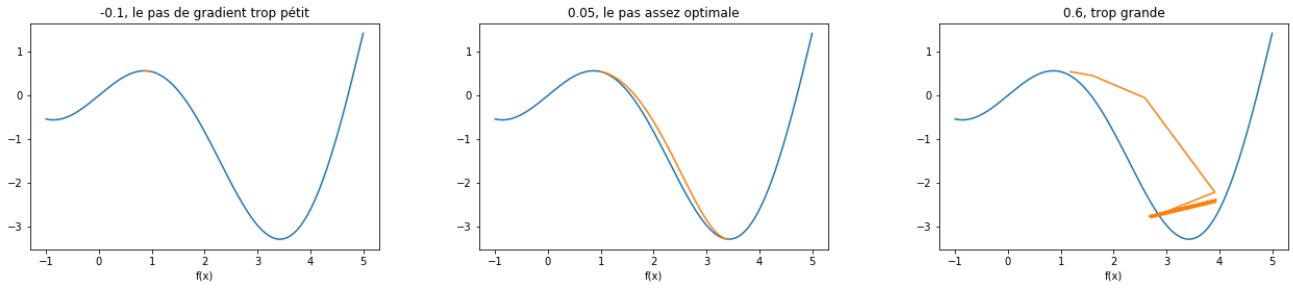
3 TME 3 : Descente de gradient

Cette semaine nous avons implémente la descente de gradient et appliqué ça à régression logistique.

3.1 Optimisation de fonctions

Dans le Figure 6 il y a des visualisations de fonctionnement de descente de gradient. Nous optimisons le fonction f donnée, en cherchent son minimum (locale, car f(x) = x cos(x) n'est pas convexe). L'axe y correspond la valeur de f(x), donc le minimum local est au fond de courbe et l'axe x correspond les valeurs de x.

FIGURE 6 – La descente de gradiente de f(x) = x cos(x)



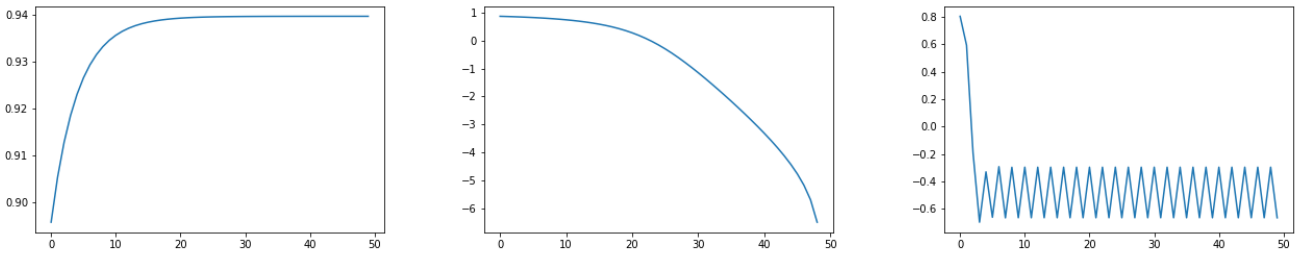
(a) (b) (c)

Si le bas de descente de gradient est trop petit ou grande, il y a une risque que, à la fin des itérations d'algorithme, nous n'arrivons pas à minimum de fonction. (Figure 6 a et c).

Dans le Figure 7 j'ai plotté le direction de descente (defini dans Équation 2) de chaque epsilon de figure precedent. On peut deduire, que le premier ne descend pas, car son epsilon est negative (direction faux). Troisiéme descend jusqu'a certain moment, mais apres il commence bugger (epsilon tros grand). Au milieu, le descente est par contre assez direct vers le minimum.

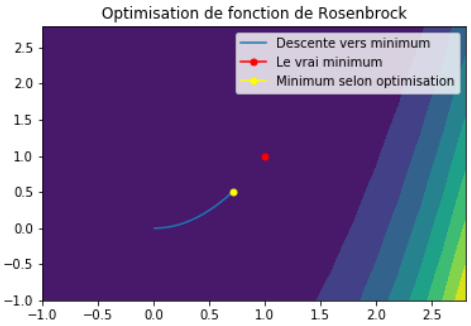
$$\left(t, \log \left\|x^t - x^*\right\|\right) \tag{2}$$

FIGURE 7 – Direction de descente (Équation 2)



Mon implémentation de descente de gradient ne fonctionne pas parfaitement. Dans le Figure 8 il y a une optimisation de fonction de Rosenbrock, où on peut voir que l'algorithme ne converge pas à vrai minimum de fonction. Cette fonction est connue pour ça que c'est difficile de converger à son minimum.

FIGURE 8 – C'est difficile de converger à minimum de fonction de Rosenbrock



3.2 Regression logistique

J'ai implémente[1] régression logistique, qui utilise la descente de gradient pour optimiser log vraisemblance.

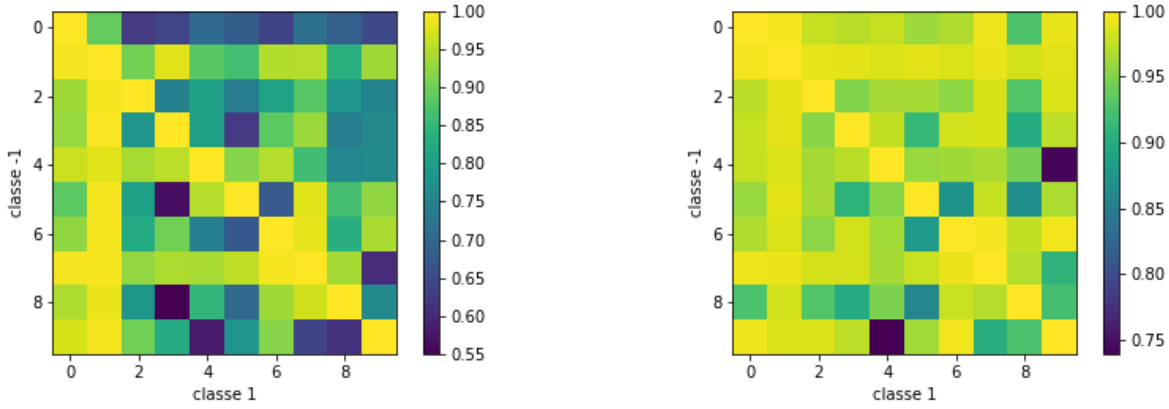
Je vais comparer classification des nombres différents (les données USPS). Pour le pas de descente de gradient j'ai fixé eps = 0.025 qui fonctionne bien avec différents nombres.

Pour chaque combination

$$(i, j) \in \{0...9\} \times \{0...9\}$$

nous calculons dans Figure 9 a) les scores avec cinq iteration de validation croisée. Puis dans Figure 9 b) il y a le precision (avec validation croisée de cinq scores) de chaque nombre comparée à tous les autres nombres. Ça va dire, que nous mettons le classe -1 pour cet nombre et classe 1 pour tous les autres.

FIGURE 9 – Les nombres un contre un



(a) Regression logistique

(b) Naif Bayes

Nous pouvons constater dans Figure 9 a) par exemple que le nombre 5 étant classe -1 et 3 étant classe 1 on obtiens score mal. Également le nombre zéro est difficile de classifier, comme on peut voir dans Figure 9 a) sur le premier ligne et dans Figure 9 b) la valeur de 0 sur axis x.

1. Code dans TME3/Logistic_regression.py

La difficulté avec zéro est probablement à cause de fait, qu'on a beaucoup plus de zeros dans les données que les autres et la forme de zero est distribué largement et est donc facile de mélanger avec les autres.

4 TME 4 et 5 : Perceptron

5 TME 5 : Support Vector Machine

5.1 Pistes de recherche

Références

A rule-of-thumb bandwidth estimator. Kernel density estimation — Wikipedia, the free encyclopedia, 2018. URL `https://en.wikipedia.org/wiki/Kernel_density_estimation#A_rule-of-thumb_bandwidth_estimator`. [Enligne ; consulté 20 Mars 2018].