Une étude sur la paramétrisation des Forêts Aléatoires

Simon Bernard, Laurent Heutte, Sébastien Adam

Une Étude sur la Paramétrisation des Forêts Aléatoires

Simon Bernard, Laurent Heutte, Sébastien Adam

Université de Rouen, LITIS EA 4108
BP 12 - 76801 Saint-Etienne du Rouvray, France.

{simon.bernard,laurent.heutte,sebastien.adam}@univ-rouen.fr

Résumé : Dans cet article nous présentons nos travaux sur la paramétrisation des Forêts Aléatoires (RF pour Random Forest), et plus précisément sur la paramétrisation de l'algorithme de référence Forest-RI. Dans cet algorithme, la "quantité" d'aléatoire injectée dans le processus d'induction d'un arbre est contrôlée par un hyperparamètre, noté K, qui joue a priori un rôle important pour construire un classifieur performant de type RF. Jusqu'à présent pourtant, aucune règle de paramétrisation n'a été proposée dans la littérature et seulement certaines valeurs arbitraires de K sont traditionnellement utilisées lorsque Forest-RI est expérimenté, sans justification théorique. Nous présentons donc dans cet article une étude expérimentale sur cet hyperparamètre, qui a pour but de comprendre dans quelle mesure le choix de K agit sur les performances en classification des RF. Nous comparons notamment entre elles les valeurs de K de la littérature à l'aide d'un test statistique de significativité. Nous comparons également ces valeurs à un nouvel algorithme d'induction de RF, appelé Forest-RK, pour lequel la valeur de K est choisie aléatoirement à chaque noeud de l'arbre, et n'est donc plus un hyperparamètre de la procédure d'induction. Nous montrons que quelques unes de ces valeurs particulières de K proposées dans la littérature sont statistiquement proches de l'optimalité sur la majorité des bases de données testées, à l'exception de certaines pour lesquelles elles sont significativement sous-optimales. Pour ces bases en particulier Forest-RK se montre en revanche au moins aussi performant que Forest-RI avec les valeurs usuelles de K. Nous reportons aussi des expérimentations complémentaires qui mettent en évidence le rôle important joué par le caractère discriminant des caractéristiques pour déterminer la valeur optimale de K.

Mots-clés : Apprentissage Supervisé, Méthode Ensembliste, Forêts Aléatoires, Arbres de Décision.

1 Introduction

Les Forêts Aléatoires (RF pour Random Forest) forment une famille de méthodes d'induction d'ensembles de classifieurs qui utilisent l'aléatoire pour générer un ensemble divers de classifieurs élémentaires. Cette famille de méthodes est définie comme

un principe générique de combinaison de classifieurs, basé sur l'utilisation de L arbres de décision en tant que classifieurs de base, notés $\{h(x, \Theta_k),\ k = 1, ...L\}$, où $\{\Theta_k\}$ est une famille de vecteurs aléatoires, indépendants et identiquement distribués, et où x représente une donnée à classer. La particularité de cet ensemble est que chacun de ces arbres de décision est construit à partir d'un vecteur aléatoire de paramètres. On espère ainsi produire la diversité nécessaire à la construction de forêts performantes.

Parmi les différents algorithmes d'induction de RF, l'algorithme Forest-RI présenté dans Breiman (2001) est souvent cité dans la littérature comme la méthode d'induction de référence. Il utilise deux principes de "randomisation" : le *Bagging* et le *Random Feature Selection*. L'étape d'apprentissage consiste donc à construire un ensemble d'arbres de décision, chacun entraîné à partir d'un sous-ensemble "bootstrap" issu de l'ensemble d'apprentissage original — *i.e.* en utilisant le principe de Bagging (Breiman, 1996) — et à l'aide d'une méthode d'induction d'arbres appelée Random Tree. Cet algorithme d'induction, habituellement basé sur l'algorithme CART (Breiman *et al.*, 1984), modifie la procédure de partitionnement des nœuds de l'arbre, de sorte que la sélection de la caractéristique utilisée comme critère de partitionnement soit partiellement aléatoire. C'est-à-dire que pour chaque nœud de l'arbre, un sous-ensemble de caractéristiques est généré aléatoirement, à partir duquel le meilleur partitionnement est réalisé. Pour résumer, dans la méthode Forest-RI, un arbre de décision est construit selon la procédure suivante :

- Pour N données de l'ensemble d'apprentissage, N individus sont tirés aléatoirement avec remise et utilisés pour l'induction de l'arbre en question.
- A chaque nœud de l'arbre, K caractéristiques sont tirées aléatoirement sans remise, parmi lesquelles la meilleure est ensuite sélectionnée pour le partitionnement (avec $K \in [1..M]$ où M est le nombre total de caractéristiques).
- L'arbre est ainsi construit jusqu'à atteindre sa taille maximale. Aucun élagage n'est réalisé.

La construction d'une forêt est donc en partie dirigée par un hyperparamètre i.e. le nombre K de caractéristiques pré-sélectionnées pour le processus de partitionnement. Ce nombre permet d'introduire plus ou moins d'aléatoire dans l'induction d'un arbre. Pourtant, bien que cet hyperparamètre semble intuitivement important pour l'induction de classifieurs performants, aucune recherche n'a spécifiquement été dédiée à l'étude du choix de sa valeur et de son influence sur les performances.

Dans Geurts *et al.* (2006), les auteurs proposent une nouvelle méthode d'induction de RF, appelée Extras-Trees pour Extremely Randomized Tree Ensembles, qui modifie l'algorithme Forest-RI, pour accentuer l'introduction de l'aléatoire via le principe de Random Feature Selection. Ils ont ensuite conçu un protocole expérimental pour étudier l'influence de K sur les performances. Même si cette méthode est partiellement différente de l'algorithme Forest-RI, ce travail permet de dresser quelques conclusions sur le comportement des Forêts Aléatoires en fonction de K. Ces travaux mettent en évidence par exemple que la valeur par défaut qu'ils ont choisie pour K ($K = \sqrt{M}$), qui est l'une des valeurs habituellement rencontrées dans la littérature) est la plupart du temps proche du paramétrage optimal, du moins pour la méthode des Extras-Trees.

Lorsqu'il introduit le formalisme des RFs dans Breiman (2001), Breiman mène quelques expérimentations pour en étudier les performances en fonction du paramètre K.

Pour ces expérimentations, un grand nombre de RFs ont été construites sur trois bases de données de l'UCI repository, pour lesquelles il s'est intéressé au taux d'erreur en classification. Parmi ces trois expérimentations, une seule concerne réellement l'algorithme Forest-RI, puisque les deux autres ont été menées sur un deuxième algorithme pour lequel une combinaison linéaire de plusieurs caractéristiques est utilisée comme critère de partitionnement, au lieu d'une caractéristique seule. Par conséquent, il est difficile d'en tirer des conclusions sur le comportement des RFs en fonction de la valeur de K. Nous avons également noté qu'au cours de ces différents tests sur Forest-RI, Breiman a choisi d'utiliser deux valeurs de K : 1 et $integer(log_2 M + 1)$. Alors que la première de ces deux valeurs est intuitivement intéressante, la seconde valeur semble un peu plus arbitraire et n'est du moins pas justifiée.

Dans Bernard *et al.* (2007), une série d'expériences a été menée avec Forest-RI, sur la base de données MNIST (LeCun *et al.*, 1998). En conclusion de ces travaux, un intervalle de valeurs de K a été trouvé pour lequel les meilleurs taux d'erreur en classification ont été obtenus ; cet intervalle ne contient ni la valeur $K = 1$, ni $K = M$ mais contient par contre les deux valeurs particulières $K = \sqrt{M}$ et $K = log_2(M) + 1$. Cependant nous pensons que ces premières conclusions nécessitent d'être confirmées en menant des expérimentations plus rigoureuses sur d'autres bases d'apprentissage.

En résumé, bien que l'implémentation et l'expérimentation de cet algorithme Forest-RI nécessitent de fixer la valeur de l'hyperparamètre K, il n'existe à ce jour aucun élément théorique ou empirique permettant de correctement choisir cette valeur. Seules quelques valeurs arbitraires sont proposées dans la littérature sans que rien ne garantisse qu'elles soient proches des valeurs optimales. Une alternative à ces valeurs de K a été proposée dans Bernard *et al.* (2008), qui consiste en un nouvel algorithme d'induction de forêts aléatoires, appelé Forest-RK, basé sur Forest-RI mais pour lequel la valeur de K est complètement choisie aléatoirement pour chaque nœud, de telle sorte que K n'est plus un hyperparamètre de la méthode.

Un des buts du travail présenté ici est donc d'étudier la paramétrisation de Forest-RI, en réalisant des comparaisons statistiques des classifieurs obtenus avec chacune des valeurs possibles de K, ainsi que des classifieurs obtenus avec Forest-RK. Nous montrons dans un premier temps que dans la plupart des cas les solutions de paramétrisation pour Forest-RI trouvées dans la littérature sont proches de la solution "optimale", tout comme Forest-RK, mais nous montrons également que pour quelques cas particuliers Forest-RK permet d'obtenir des classifieurs plus performants alors que les autres solutions sont significativement sous-optimales. Nous proposons également avec ces travaux une analyse des mécanismes de fonctionnement des RF en étudiant la relation entre la nature de l'espace de caractéristiques et les performances. Nous mettons en évidence le rôle crucial joué par la "qualité" des caractéristiques dans la recherche de la meilleure valeur de K.

2 Étude du choix de l'hyperparamètre K

L'hyperparamètre K fixe le nombre de caractéristiques sélectionnées aléatoirement à chaque nœud au cours de la procédure d'induction d'un arbre. Sa valeure est donc choisie dans l'intervalle $[1..M]$, où M représente la dimension de l'espace de description.

Ce nombre contrôle donc la quantité d'aléatoire introduit dans le processus de sélection de caractéristiques, de telle sorte que plus la valeur de K est petite et plus on introduit d'aléatoire. Dans le cas où $K = 1$ par exemple, chaque critère de partitionnement de l'arbre est choisi entièrement aléatoirement parmi les caractéristiques disponibles. À l'inverse, lorsque $K = M$, l'aléatoire n'intervient pas dans la sélection du critère de partitionnement, et chaque arbre est donc construit à l'aide d'une procédure d'induction classique. Dans ce cas particulier l'aléatoire est introduit à l'aide de la méthode de Bagging uniquement.

L'idée principale de nos expérimentations est de comparer différentes valeurs de K traditionnellement utilisées dans la littérature, pour percevoir si l'une ou l'autre de ces valeurs est proche de l'optimalité en termes d'erreur en généralisation. Dans la sous-section suivante, nous décrivons dans un premier temps les bases de données utilisées dans nos expérimentations, puis dans les deux sous-sections qui suivent nous détaillons le protocole expérimental mis en œuvre, ainsi que les résultats obtenus.

2.1 Description des bases de données

Une description des 12 bases de données qui ont été utilisées au cours de ces expérimentations est donnée dans le tableau 1. Neuf de ces bases de données ont été sélectionnées à partir de l'UCI repository (Asuncion & Newman, 2007), principalement parce qu'elles présentent une large variété de problèmes d'apprentissage automatique en termes de nombre de classes, de nombre de caractéristiques et de nombre de données. Trois bases de données supplémentaires concernant des problèmes de reconnaissance de chiffres manuscrits, ont également été utilisées : (i) MNIST (LeCun *et al.*, 1998), de laquelle ont été extraites 85 caractéristiques comme expliqué dans Bernard *et al.* (2007) ; (ii) Digits et DigReject, toutes deux décrites dans Chatelain *et al.* (2006).

TABLE 1 – Description des bases de données

Bases de données	# Données	# Caractéristiques	# Classes
Digits	38142	330	10
DigReject	14733	330	2
Letter	20000	16	26
Madelon	2600	500	2
Mfeat-factors	2000	216	10
Mfeat-fourier	2000	76	10
Mfeat-karhunen	2000	64	10
Mfeat-zernike	2000	47	10
MNIST	60000	84	10
Musk	6597	166	2
Pendigits	10992	16	10
Segment	2310	19	7

2.2 Protocole expérimental

Il s'agit d'effectuer des tests comparatifs sur les bases de données présentées dans le tableau 1. Pour toutes les RF induites au cours de nos expériences, le nombre d'arbres a été fixé à 100. Ce choix est basé sur un travail expérimental présenté dans Bernard *et al.* (2007), dans lequel il est montré que c'est une valeur raisonnable pour construire des RF performantes. En outre, notre but n'est pas ici d'obtenir les meilleures performances possibles mais plutôt de comparer les performances relatives des RF induites.

Tout d'abord, les bases de données utilisées n'étant pas préalablement divisées en données d'apprentissage et de test, chacune de ces bases a été aléatoirement séparée en deux sous-ensembles. Cette procédure de découpage a été répétée 50 fois, de sorte que nous disposons de 50 ensembles d'apprentissage et de test differents, chacun contenant respectivement deux tiers et un tiers des données. Nous notons ces découpages $T_i = (Tr_i, Ts_i)$ avec $i \in [1..50]$ et où Tr_i et Ts_i représentent respectivement l'ensemble d'apprentissage et de test. Ensuite pour chaque T_i, l'algorithme Forest-RI a été lancé pour chaque valeur de K dans l'intervalle $[1..M]$, où M représente le nombre total de caractéristiques. Enfin, l'algorithme Forest-RK a été lancé sur chaque T_i.

Notre problème est donc de déterminer lequel des algorithmes, parmi Forest-RI avec les différentes valeurs d'hyperparamètre K et Forest-RK, permettra d'obtenir les classifieurs les plus performants lorsqu'ils sont entrainés et testés sur les mêmes bases de données. Pour répondre à cette problématique, nous nous appuyons sur une étude comparative de cinq tests statistiques présentée dans Dietterich (1998). Dans cet article il est montré que le test qui correspond le mieux aux protocoles expérimentaux comme le nôtre est le test de McNemar (Everitt, 1977). Ce test est utilisé ici pour comparer deux algorithmes sur la base de leurs prédictions pour un ensemble de données de test. Il nous permet par conséquent d'obtenir trois réponses possibles :

- l'algorithme B produit des classifieurs significativement plus performants que l'algorithme A.
- l'algorithme A produit des classifieurs significativement plus performants que l'algorithme B.
- les deux algorithmes produisent des classifieurs qui ne sont pas significativement différents en terme de taux d'erreurs.

Avec une telle procédure, nous sommes alors capables de déterminer lequel des deux algorithmes d'apprentissage surpasse l'autre statistiquement, en terme de taux d'erreur. De plus en l'appliquant à chacun des $50 T_i$, il nous permet d'avoir une idée de la variabilité des résultats de ces comparaisons entre chaque paire d'algorithmes. L'algorithme 1 décrit le protocole expérimental complet appliqué à chaque base de données. Les résultats qu'il nous a permis d'obtenir sont présentés et discutés dans la sous-section suivante.

2.3 Résultats

Le tableau 2 présente une synthèse des résultats obtenus avec le protocole expérimental détaillé dans la section 2.2. Il permet donc d'obtenir un tableau de $50 \times M$ taux d'erreur en fonction de K pour chaque base de données. Ces séries de taux d'erreur ont été moyennées de sorte d'obtenir M valeurs moyennes pour chaque base de don-

Algorithme 1 Protocole Expérimental

ENTRÉE: N le nombre de données disponibles.
ENTRÉE: M le nombre de caractéristiques disponibles.
SORTIE: $\epsilon_{RI}[50][M']$ un tableau 2D pour conserver les taux d'erreur obtenus avec Forest-RI.
SORTIE: $\epsilon_{RK}[50]$ un tableau 1D pour conserver les taux d'erreur obtenus avec Forest-RK.
SORTIE: $\mathcal{M}[50][M'+1][M'+1]$ un tableau 3D pour conserver les réponses au test de McNemar.
pour $i \in [1..50]$ **faire**
  Tirer aléatoirement sans remise $\frac{2}{3} \times N$ des données disponibles pour former l'ensemble d'apprentissage Tr_i. Les données restantes forment alors l'ensemble de test Ts_i, le couple (Tr_i, Ts_i) est noté T_i.
  pour chaque valeur k de $[1..M]$ **faire**
    $H_{RI}(k) \leftarrow$ Induire une RF avec l'algorithme Forest-RI, avec $L = 100$ et $K = k$, sur Tr_i.
    $\epsilon_{RI}(i,k) \leftarrow$ Tester la RF résultante sur Ts_i.
  fin pour
  $h_{RK} \leftarrow$ Induire une RF avec l'algorithme Forest-RK sur Tr_i.
  $\epsilon_{RK}(i) \leftarrow$ Tester la RF résultante sur Ts_i.
  pour $(h_m, h_n) \in H_{RI} \times H_{RI}$ **faire**
    $\mathcal{M}(i, h_m, h_n) \leftarrow$ Lancer le test de McNemar avec les classifieurs (h_m, h_n) sur Ts_i.
  fin pour
  pour $h_n \in H_{RI}$ **faire**
    $\mathcal{M}(i, h_{RK}, h_n) \leftarrow$ Lancer le test de McNemar avec les classifieurs (h_{RK}, h_n) sur Ts_i.
  fin pour
fin pour

nées, *i.e.* une valeur moyenne pour chaque K ; les valeurs d'écarts-types ont aussi été calculées. Le tableau 2 détaille quelques uns de ces résultats concernant quatre paramétrisations particulières pour K. La première de ces valeurs, dans la seconde colonne, correspond à la valeur "optimale", notée K^*, c'est-à-dire la valeur de K parmi toutes les valeurs testées pour laquelle le taux d'erreur moyen minimum a été obtenu. Les seconde et troisième valeurs correspondent respectivement à $K = \sqrt{M}$ et $K = log_2 M + 1$. La dernière colonne de valeurs correspond aux taux d'erreur moyens obtenus avec l'algorithme Forest-RK (RK dans le tableau 2). Les nombres entre parenthèses pour les colonnes 2,3 et 4, représentent les valeurs de K qui ont été soit reportées de nos expérimentations (K^* dans la colonne 2) soit fixées ($\sqrt{M}$ dans la colonne 3 ou $log_2(M) + 1$ dans la colonne 4). A noter que la valeur $K = 1$ n'est pas reportée dans ce tableau car, comme le montre le tableau 3, elle est clairement sous-optimale.

Une première observation faite à partir du tableau 2 est que les deux valeurs particulières $K = \sqrt{M}$ et $K = log_2(M) + 1$ correspondent rarement exactement à la valeur

TABLE 2 – Taux d'erreur obtenus pour les différents algorithmes.

Dataset	K^*	$K_{\sqrt{M}}$	$K_{log_2(M)+1}$	RK
Digits	2.18 ± 0.12 (11)	2.20 ± 0.13 (18)	2.19 ± 0.12 (9)	2.60 ± 0.11
DigRej	7.15 ± 0.34 (181)	7.70 ± 0.34 (18)	7.80 ± 0.35 (9)	7.16 ± 0.33
Letter	4.16 ± 0.28 (3)	4.23 ± 0.23 (4)	4.30 ± 0.26 (5)	4.42 ± 0.29
Madelon	17.60 ± 1.60 (261)	30.48 ± 1.94 (22)	34.54 ± 1.26 (10)	18.88 ± 1.60
Mfeat-fa	3.56 ± 0.71 (10)	3.57 ± 0.58 (15)	3.58 ± 0.73 (9)	3.97 ± 0.81
Mfeat-fo	16.81 ± 1 (19)	17.11 ± 1.05 (9)	17.25 ± 1.02 (7)	17.44 ± 1.22
Mfeat-ka	4.30 ± 0.68 (6)	4.33 ± 0.69 (7)	4.30 ± 0.68 (6)	6.20 ± 0.80
Mfeat-ze	22.26 ± 1.06 (7)	22.26 ± 1.06 (7)	22.56 ± 1.02 (5)	22.82 ± 1.43
MNIST	5.06 ± 0.14 (21)	5.17 ± 0.14 (10)	5.32 ± 0.17 (8)	5.18 ± 0.14
Musk	2.34 ± 0.34 (88)	2.40 ± 0.29 (13)	2.50 ± 0.26 (8)	2.45 ± 0.29
Pendig	0.97 ± 0.17 (4)	0.97 ± 0.17 (4)	1.01 ± 0.17 (5)	1.01 ± 0.19
Segm	2.36 ± 0.53 (5)	2.44 ± 0.44 (4)	2.36 ± 0.53 (5)	2.32 ± 0.57

de K^*. K^* est égal à $\sqrt{M}$ pour seulement 2 des 12 bases de données (Mfeat-zernike et Pendigits) et à $log_2(M)+1$ également pour seulement 2 d'entre elles (Mfeat-karhunen et Segment). Ces valeurs sont même parfois très éloignées de la valeur de K^*, à l'image des bases de données Madelon, DigRejects, MNIST, et Musk. On pourrait en conclure qu'il n'est pas conseillé de systématiquement utiliser l'une ou l'autre de ces valeurs de K et qu'il serait alors nécessaire d'étudier un moyen de fournir une règle de paramétrisation plus précise. Cependant, en examinant les valeurs d'écarts-types et les différences entre deux résultats obtenus pour deux valeurs de K proches l'une de l'autre, nous nous demandons si cette conclusion est correcte. C'est la raison pour laquelle nous avons décidé d'utiliser le test de McNemar dans le but de nous rendre compte plus précisément si ces différences de performances sont statistiquement significatives. Les résultats de ce test sont reportés dans le tableau 3, à l'aide des nombres de cas pour lesquels chacune des trois réponses possibles a été obtenue. Pour chaque base de données et pour chaque paire d'algorithmes on dispose donc de 50 réponses au test réparties selon les 3 cas possibles. De cette façon il est aisé de se rendre compte que pour une grande majorité des cas, K^*, $K_{\sqrt{M}}$ et $K_{log_2(M)+1}$ permettent d'obtenir des performances statistiquement comparables. Seule la valeur $K = 1$ ne permet pas d'obtenir des classifieurs proches de l'optimalité. De plus, la même remarque globale peut être faite pour l'algorithme Forest-RK, *i.e.* il permet dans la plupart des cas d'induire des classifieurs statistiquement aussi performants que Forest-RI avec K^*. Tous ces résultats tendent à montrer que pour la plupart des problèmes d'apprentissage automatique, il n'est pas crucial de fournir une régle de paramétrisation qui permettrait d'améliorer les performances en comparaison avec les valeurs traditionnellement utilisées.

Il n'en reste pas moins intéressant d'essayer de comprendre les mécanismes de fonctionnement des RF et la façon dont cet hyperparamètre K agit sur les performances. La figure 1 présente nos résultats sous la forme de courbes de taux d'erreur moyens en fonction de la valeur de K. Sur cette figure on peut tout d'abord observer que toutes les courbes suivent la même variation globale, c'est-à-dire une partie décroissante suivie

d'une partie croissante. Cela confirme les premières conclusions obtenues dans Bernard *et al.* (2007), où il a été montré que les extrema pour K, *i.e.* $K = 1$ et $K = M$, ne sont pas conseillés pour induire des classifieurs performants avec Forest-RI. Cependant, à un niveau de détail plus précis, ces variations de courbes semblent devenir très différentes d'une base à une autre. On peut distinguer trois grandes tendances de variation pour ces 12 diagrammes : pour 5 d'entre elles (Mfeat-factors, Mfeat-fourier, Mfeat-karhunen, Mfeat-zernike et Digits) le taux d'erreur minimum est atteint pour une valeur très faible de K (marqué d'un cercle sur la figure) et l'augmentation des valeurs à partir de ce point jusqu'à $K = M$ est monotone et presque linéaire ; pour 4 d'entre elles (Letter, Mnist, Pendigits et Segment) cette tendance est presque identique mais avec une forme un peu plus parabolique ; pour 3 d'entre elles enfin (DigReject, Madelon and Musk) il n'y a pratiquement pas de partie croissante des valeurs et le taux d'erreur minimum est atteint pour une valeur bien plus grande de K (*i.e.* supérieure à $\frac{M}{2}$). Ces différents comportements sont relativement difficiles à expliquer uniquement sur la base des propriétés des bases de données telles que le nombre de caractéristiques ou le nombre de classes. Dans Geurts *et al.* (2006), les auteurs suggèrent que la nature des caractéristiques pourrait expliquer les comportements particuliers qu'ils ont constatés avec leurs Extra-Trees sur certaines bases de données. Ils conjecturent que plus il y a de caractéristiques discriminantes et plus la valeur de K^* est élevée, puisqu'une grande valeur pour K donne de meilleures chances de filtrer les caractéristiques non discriminantes. Cette affirmation nous a conduit à nous intéresser à la qualité des caractéristiques pour expliquer les variations de performances en fonction de K. Pour ce faire, nous avons décidé d'évaluer la qualité de chaque caractéristique en mesurant son gain d'information. De façon générale, le gain d'information mesure la variation d'entropie entre un état initial et un état après l'apport d'information (Guyon & Elisseeff, 2003). Cette mesure est souvent utilisée pour le choix du critère de partitionnement dans les arbres de décision. Pour nos expérimentations, la valeur du gain d'information a été mesurée pour toutes les caractéristiques sur chacun des Tr_i de chaque base de données. $50 \times M$ valeurs de gain d'information ont donc été calculées pour chaque base. La figure 2 synthétise ces résultats sous la forme de courbes de nombres cumulés de caractéristiques en fonction des valeurs de gain d'information, de telle sorte que chaque point de ces courbes indique le nombre de caractéristiques pour lesquelles le gain d'information est inférieure ou égal à la valeur correspondante sur l'axe des abscisses. Cette représentation permet de visualiser simultanément la qualité des caractéristiques ainsi que le nombre d'entre elles qui ne sont pas discriminantes. On peut observer sur cette figure que les valeurs de gain d'information sont globalement grandes (typiquement supérieures à 0.1) pour les bases de données pour lesquelles la valeur de K^* est petite par rapport à M, comme c'est le cas par exemple pour Digits, Mfeat-factors et Mfeat-Karhunen pour lesquelles K^* est plus petit que $\frac{M}{10}$. A l'inverse, pour les trois bases de données pour lesquelles K^* est plus grand que $\frac{M}{2}$ (DigReject, Madelon et Musk), le gain d'information est toujours inférieur à environ 0.1. Cela semble prouver que l'information intrinséque apportée par les caractéristiques explique très fortement les variations de performances en fonction de K.

Le choix de K permet en fait d'établir un compromis entre deux besoins : (i) forcer, via l'aléatoire, le processus d'induction d'arbre à diversifier les choix de critère de

partitionnement dans le but de créer des arbres différents les uns des autres, (ii) choisir des caractéristiques discriminantes comme critère de partitionnement dans le but d'induire des arbres suffisamment performants. Trop d'aléatoire injecté dans la sélection du critère de partitionnement produit des arbres qui globalement ne sont pas adaptés au problème, alors que ne pas injecter suffisamment d'aléatoire crée des arbres qui tendent à sur-apprendre, et donc à être relativement similaire les uns aux autres en terme de prédictions. K permet donc d'établir un juste compromis entre les performances individuelles des arbres, et la diversité dans l'ensemble. Cependant nous pensons que l'impact sur les performances de la "quantité" d'aléatoire injecté dans le processus de sélection du critère de partitionnement, est fortement modifiée par la qualité globale des caractéristiques. S'il y a trop peu de caractéristiques discriminantes, l'aléatoire va rapidement provoquer une détérioration des performances, et donc rapidement détériorer le compromis "performance individuelle des arbres contre diversité de l'ensemble". À l'inverse, une quantité importante de caractéristiques fortement discriminantes va faciliter le surapprentissage des arbres et donc affaiblir l'effet de l'aléatoire dans la sélection du partitionnement. Nous pensons que c'est la raison pour laquelle les trois bases de données, pour lesquelles le gain d'information est faible (DigReject, Madelain et Musk), présentent des courbes de taux d'erreur qui n'augmentent pas significativement pour des valeurs croissantes de K. Par conséquent nous pensons que l'information apportée par les caractéristiques est une propriété importante à prendre en compte pour déterminer une régle de paramétrisation pour l'algorithme Forest-RI.

3 Conclusions

Nous nous sommes intéressés dans cet article au nombre K de caractéristiques sélectionnées aléatoirement à chaque nœud d'un arbre d'une RF. Cet hyperparamètre permet de contrôler la quantité d'aléatoire injecté pour la sélection du partitionnement, de telle sorte que plus la valeur de K est petite et plus l'aléatoire intervient dans l'induction de l'arbre.

Nous avons tout d'abord montré que les différentes solutions de paramétrisation trouvées dans la littérature pour l'algorithme de RF Forest-RI, permettent d'induire des classifieurs performants, en ce sens qu'ils sont proches des performances obtenues avec le meilleur K, que nous notons K^*. Dans la majorité des cas, les valeurs $K = \sqrt{M}$ et $K = log_2(M) + 1$ se sont montrées comparables aux résultats obtenus avec la valeur K^*, d'après le test de significativité de McNemar. En outre, aucune de ces deux valeurs n'a présenté de meilleures performances que l'autre à l'exception d'une base de données pour laquelle $K = \sqrt{M}$ est un meilleur choix. $K = 1$, en revanche, ne permet jamais d'induire un classifieur suffisamment performant en comparaison avec K^*. Pour quelques cas particuliers seulement, les RF ont présenté un comportement différent, à savoir que les trois valeurs $K = \sqrt{M}$, $K = log_2(M) + 1$ et $K = 1$ n'ont pas permis d'obtenir des performances comparables à celles obtenues avec K^*. D'un autre coté, Forest-RK, pour lequel la valeur de K est choisie aléatoirement à chaque nœud, a offert des performances comparables à Forest-RI avec K^*, comme c'est le cas pour la plupart des autres bases de données.

Nous pensons que ces comportements particuliers peuvent être expliqués par la na-

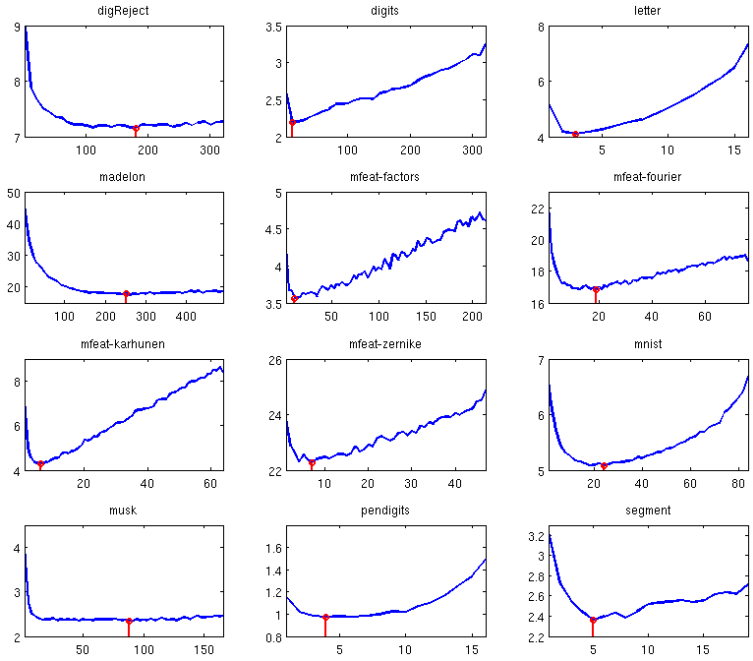

FIGURE 1 – Taux d'erreur moyens en fonction des valeurs de K. Les taux d'erreur minimum sont marqués sur chaque diagramme.

ture des caractéristiques et plus particulièrement du caractère discriminant des caractéristiques. Nous en avons donc mesuré la qualité à l'aide du gain d'information pour chacune des bases de données. Ces expérimentations ont montré que cette propriété est cruciale pour déterminer la valeur de K^*, car elle modifie fortement les effets de l'introduction de l'aléatoire via la procédure de *random feature selection*. Par conséquent nous pensons que le caractère discriminant des caractéristiques est une propriété importante à prendre en compte pour pouvoir établir une régle de paramétrisation pour l'algorithme Forest-RI. Nos travaux futurs auront pour but de répondre à cette problématique.

Références

ASUNCION A. & NEWMAN D. (2007). UCI machine learning repository.

BERNARD S., HEUTTE L. & ADAM S. (2007). Using random forests for handwritten digit recognition. *International Conference on Document Analysis and Recognition*, p. 1043–1047.

BERNARD S., HEUTTE L. & ADAM S. (2008). Forest-rk : A new random forest induction method. *International Conference on Intelligent Computing*, p. 430–437.

BREIMAN L. (1996). Bagging predictors. *Machine Learning*, **24**(2), 123–140.

BREIMAN L. (2001). Random forests. *Machine Learning*, **45**(1), 5–32.

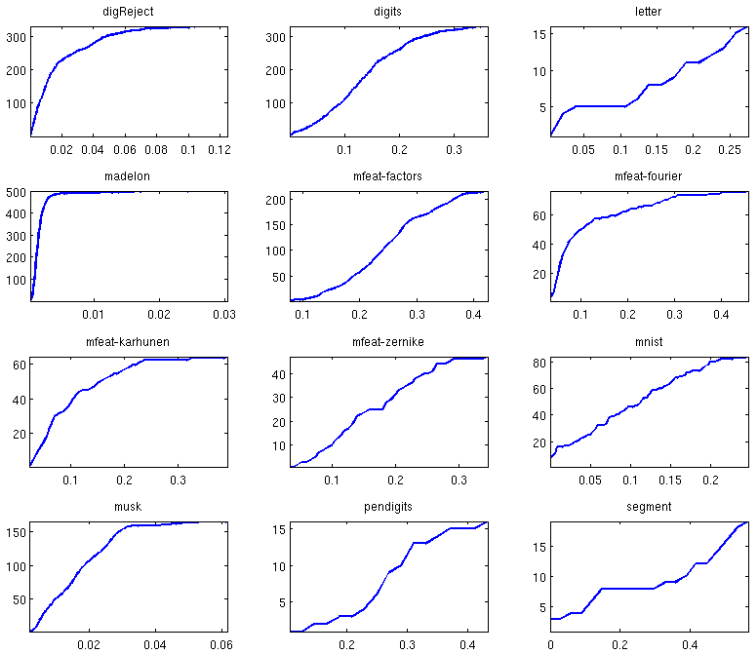

FIGURE 2 – Nombre cumulé de caractéristiques en fonction des valeurs de gain d'information.

BREIMAN L., FRIEDMAN J., OLSHEN R. & STONE C. (1984). *Classification and Regression Trees*. Chapman and Hall (Wadsworth, Inc.) : New York.

CHATELAIN C., HEUTTE L. & PAQUET T. (2006). A two-stage outlier rejection strategy for numerical field extraction in handwritten documents. *Internation Conference on Pattern Recognition, Honk Kong, China*, **3**, 224–227.

DIETTERICH T. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, **10**, 1895–1923.

EVERITT B. (1977). *The Analysis of Contingency Tables*. Chapman and Hall, London.

GEURTS P., ERNST D. & WEHENKEL L. (2006). Extremely randomized trees. *Machine Learning*, **36**(1), 3–42.

GUYON I. & ELISSEEFF A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, **3**, 1157–1182.

LECUN Y., BOTTOU L., BENGIO Y. & HAFFNER P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**(11), 2278–2324.

TABLE 3 – Résultats du test de McNemar. Dans chaque cellule le premier nombre correspond aux cas pour lesquels l'algorithme A est plus performant que l'algorithme B ; le second aux cas pour lesquels les deux algorithmes sont statistiquement équivalents ; et le troisième nombre aux cas pour lesquels B est meilleur que A.

Algo A	K^*			$K_{\sqrt{M}}$		K_1	RK			
Algo B	K_1	$K_{\sqrt{M}}$	K_{log}	K_1	K_{log}	K_{log}	K^*	K_1	$K_{\sqrt{M}}$	K_{log}
Digits	**48**/2/0	2/**48**/0	1/**49**/0	**48**/2/0	0/**50**/0	0/0/**50**	0/1/**49**	4/**46**/0	0/1/**49**	0/4/**46**
DigReject	**50**/0/0	**25**/**25**/0	**37**/13/0	**50**/0/0	13/**37**/0	0/2/**48**	2/**46**/2	**50**/0/0	**27**/23/0	**41**/9/0
Letter	**50**/0/0	3/**45**/2	3/**47**/0	**50**/0/0	4/**46**/0	0/1/**49**	0/**33**/17	**46**/4/0	0/**34**/16	0/**30**/10
Madelon	**50**/0/0	**50**/0/0	**50**/0/0	**50**/0/0	**45**/5/0	0/0/**50**	0/**43**/7	**50**/0/0	**50**/0/0	**50**/0/0
Mfeat-fac	4/**46**/0	0/**49**/1	0/**49**/1	4/**46**/0	0/**50**/0	0/**43**/7	0/**48**/2	3/**47**/0	0/**46**/4	0/**47**/3
Mfeat-fou	**47**/3/0	1/**49**/0	1/**49**/0	**46**/4/0	1/**49**/0	0/2/**48**	0/**47**/3	**42**/8/0	0/**45**/5	1/**47**/2
Mfeat-kar	**45**/5/0	2/**47**/1	3/**46**/1	**42**/8/0	1/**49**/0	0/9/**41**	0/**35**/15	2/**48**/0	0/16/**34**	0/21/**29**
Mfeat-zer	7/**43**/0	0/**49**/1	1/**48**/1	7/**43**/0	1/**48**/1	0/**40**/10	0/**49**/1	4/**46**/0	0/**49**/1	0/**50**/0
MNIST	**50**/0/0	7/**43**/0	23/**27**/0	**50**/0/0	6/**43**/1	0/0/**50**	0/**39**/11	**50**/0/0	2/**47**/1	4/**46**/0
Musk	**50**/0/0	1/**49**/0	6/**44**/0	**50**/0/0	4/**46**/0	0/0/**50**	0/**48**/2	**50**/0/0	2/**48**/0	5/**44**/1
Pendigits	6/**44**/0	0/**49**/1	0/**49**/1	5/**45**/0	1/**49**/0	0/**43**/7	0/**50**/0	3/**47**/0	0/**50**/0	2/**48**/0
Segment	10/**40**/0	1/**48**/1	0/**50**/0	9/**41**/0	0/**48**/2	0/**37**/13	1/**49**/0	13/**37**/0	0/**49**/1	1/**49**/0