Sélection supervisée d'instances : une approche descriptive

Sylvain Ferrandiz[*,**], Marc Boullé[*]

[*]France Télécom R&D,
2, avenue Pierre Marzin, 22300 Lannion
sylvain.ferrandiz@francetelecom.com,
marc.boulle@francetelecom.com,
[**]GREYC, Université de Caen,
boulevard du Maréchal Juin, BP 5186, 14032 Caen Cedex,

Résumé. La classification suivant le plus proche voisin est une règle simple et performante. Sa mise en oeuvre pratique nécessite, tant pour des raisons de coût de calcul que de robustesse, de sélectionner les instances à conserver. La partition de Voronoi induite par les prototypes constitue la structure sous-jacente à cette règle. Dans cet article, on introduit un critère descriptif d'évaluation d'une telle partition, quantifiant le compromis entre nombre de cellules et discrimination de la variable cible entre les cellules. Une heuristique d'optimisation est proposée, tirant partie des propriétés des partitions de Voronoi et du critère. La méthode obtenue est comparée avec les standards sur une vingtaine de jeux de données de l'UCI. Notre technique ne souffre d'aucun défaut de performance prédictive, tout en sélectionnant un minimum d'instances. De plus, elle ne sur-apprend pas.

1 Introduction

La classification supervisée constitue un problème d'apprentissage classique. On dispose dans ce cas, en plus des variables descriptives (ou endogènes), d'une variable cible (ou exogène). En phase d'exploration des données, c'est la dépendance de la variable cible vis-à-vis des variables descriptives qu'on vise à expliciter. En phase de modélisation, le but est de fournir la meilleure prédiction possible pour toute nouvelle instance à classifier. Quelle que soit la situation, la connaissance est à extraire d'un échantillon de N instances étiquetées.

Une méthode de classification usuelle est la règle de classification suivant le plus proche voisin introduite par Fix et Hodges (1951). Elle consiste à attribuer à une instance l'étiquette de l'instance la plus proche parmi celles constituant l'échantillon. La mise en œuvre de cette modélisation soulève deux questions fondamentales :

– Quelle mesure de similitude employer ?
– Quelles instances de l'échantillon conserver ?

La première question couvre plusieurs champs d'investigation : gestion de la présence jointe de variables continues et symboliques, normalisation des variables continues, prétraitement des variables symboliques, pondération de la contribution des variables, etc. Dans le cas continu, l'usage a consacré l'emploi de la distance euclidienne et des distances L_p ($p \geq 1$)

de Minkowski. La distance de Mahalanobis (Duda et al., 2001), en effectuant une transformation globale des instances et au prix d'un coût de calcul plus élevé, permet d'intégrer dans le calcul de la similitude les corrélations entre couples de variables descriptives. Pour des mesures successives d'une même quantité, le Dynamic Time Warping est un procédé traitant la corrélation temporelle (Berndt et Clifford, 1996). Dans le cas symbolique, la distance de Hamming est d'autant plus simplificatrice que le nombre de modalités des variables croît. C'est pourquoi des mesures de similitude basées sur les probabilités d'occurence sont souvent utilisées, par Stanfill et Waltz (1986) entre autres. Un procédé de gestion de la mixité (variables continues et symboliques) est proposé par Wilson et Martinez (1997a). On ne s'intéresse pas dans cet article à la question du choix d'une mesure de similitude et on se focalise sur la sélection d'instance.

La classification par le plus proche voisin impose, pour chaque nouvelle instance à classifier, de parcourir l'ensemble de l'échantillon. Ceci entraîne un coût de déploiement rédhibitoire. Une étape de sélection des instances permet de diminuer le coût de recherche. Classiquement, les méthodes prédictives s'attellent à qualifier le degré d'utilité prédictive d'une instance et conservent les instances jugées utiles. Ces méthodes sont décrites à la section 2.

Les instances sélectionnées (ou : prototypes) induisent une partition de Voronoi de l'espace, à chaque groupe étant associé un prototype. L'ensemble des instances se répartit dans les groupes de cette partition. On considère alors la distribution des étiquettes dans chacun des groupes. Là où les méthodes prédictives ne prennent en compte que l'étiquette du prototype , on associe à chaque prototype cette distribution de probabilité. Cette dichotomie reflète celle observée dans tout processus de fouille de données entre la phase de préparation des données et celle de modélisation (Chapman et al., 2000). On présente à la section 3 une approche descriptive de l'évaluation de la qualité de telles fonctions, ainsi que le critère qui en découle. La question du sur-apprentissage étant prise en charge par le critère, on propose à la section 4 une heuristique d'optimisation poussée. Enfin, une comparaison expérimentale sur données réelles est menée à la section 5.

2 La sélection d'instances

La méthode CNN (pour Condensed Nearest Neighbor) décrite par Hart (1968) est la plus ancienne méthode de sélection d'instances. Toute instance mal classifiée par son plus proche voisin parmi les prototypes déjà sélectionnés est aussitôt conservée. Ce procédé incrémental est itéré tant qu'il existe des instances mal classifiées par l'ensemble de prototypes. La méthode est consistante, dans le sens où tout élément de l'échantillon est bien classifié par son plus proche prototype. La complexité au pire de cet algorithme est un $O(N^3)$.

Une amélioration est proposée par Gates (1972) : RNN (pour Reduced Nearest Neighbor). Une fois la règle CNN appliquée, toute suppression d'un prototype ne provoquant aucune mauvaise classification d'une instance est validée. L'intérêt de cette méthode réside dans sa capacité à produire un sous-ensemble de prototypes de taille minimale relativement à la condition de consistance, sous réserve qu'un tel sous-ensemble soit inclus dans la solution proposée par CNN.

Les décisions prises par ce type de techniques sont peu robustes (conservation du bruit, par exemple). Afin d'y remédier, une idée consiste à prendre en compte les K plus proches voisins (typiquement, $K = 3$). Le procédé est décrémental et une instance est éliminée si elle est mal classifiée par un vote à la majorité sur ses K plus proches voisins. C'est la règle ENN, pour

Edited Nearest Neighbor, présentée par Wilson (1972). Sa complexité algorithmique est un $O(KN^2)$. L'élimination est ainsi fiabilisée mais limitée. Notons que cette méthode peut être appliquée itérativement.

Les précédentes méthodes étant de complexité élevée ou ne réalisant pas l'objectif d'une sélection drastique, d'après Wilson et Martinez (2001), Aha et al. (1991) ont proposé une série d'algorithmes dont IB3 est la version la plus aboutie. Une notion d'acceptabilité est introduite dans CNN : une instance est conservée si elle est mal classifiée par le plus proche prototype acceptable. La complexité algorithmique d'IB3 est un $O(N^2)$. Un prototype est acceptable si son taux de bonne classification est significativement supérieur à la fréquence de sa classe. De plus, sont éliminés les prototypes peu acceptables. Ceci permet de fiabiliser les décisions tout en évitant d'être trop conservateur.

Dans ce but également, la notion d'association est utilisée par Wilson et Martinez (1997b). Pour $K \leq N$ est fixé, si x est l'un des K plus proches voisins de y, on dit que y est associé à x. Autrement dit, x est associé à y s'il participe à la classification de y. Dès lors, x est éliminé si le nombre de ses associés bien classifiés ne diminue pas après sa suppression. La règle ENN est préliminairement appliquée. De plus, les instances sont considérées par ordre décroissant de distance à la plus proche instance de classe différente. La méthode obtenue, DROP3, est de complexité un $O(KN^2)$.

A la croisée des chemins entre IB3 et DROP3, on trouve un test statistique évaluant l'hypothèse de non contribution d'une instance à la classification de ses associés (Sebban et al., 2002). Ce critère est paramétrique et son calcul nécessite l'approximation de la densité de la statistique associée. Une adaptation de l'algorithme AdaBoost permettant de traiter des classifieurs locaux (les prototypes) aboutit à une heuristique de recherche incrémentale. Dans sa version la plus rapide, l'algorithme est de complexité un $O(KN^2)$.

Cameron-Jones (1995) a proposé un critère d'évaluation de la qualité prédictive d'un ensemble de prototypes. L'approche adoptée est de type MML (pour Minimum Message Length) et le critère obtenu s'écrit :

$$c(K, N, E) = F(K, N) + K \log_2(J) + F(E, N - K) + E \log_2(J - 1),$$

avec K le nombre de prototypes, N le nombre d'instances, E le nombre d'instances mal classifées par leur plus proche prototype et J le nombre de classe cibles. La quantité $F(U, V)$ mesure la longueur du mot de code nécessaire à la spécification de U instances parmi V et est évaluée par la formule :

$$F(U, V) = \log^* \left(\sum_{u=0}^{U} \binom{V}{u} \right),$$

où $\log^*(x)$ désigne la somme des termes positifs $\log_2(x)$, $\log_2(\log_2(x))$, etc. Les termes $K \log_2(J)$ et $E \log_2(J-1)$ correspondent aux longueurs de code nécessaires à la spécification des étiquettes des K prototypes et des E exceptions respectivement.

Une heuristique est également proposée, qu'on nomme ici Explore. Une première phase itérative consiste à ajouter une instance si la valeur du critère diminue. Une fois toutes les instances considérées, tout prototype dont la suppression conduit à la diminution de la valeur du critère est effectivement éliminé. Enfin, 1000 mutations sont évaluées et acceptées si la valeur du critère décroît. Une mutation est soit un ajout d'une instance à l'ensemble des prototypes, soit une suppression d'un prototype, soit un échange entre une instance et un prototype. Au final, la méthode est de complexité un $O(N^3)$ au pire et proche d'un $O(N^2)$ en moyenne.

3 Une approche descriptive

L'approche prédictive classique consiste à évaluer la qualité prédictive d'un classifieur, en mesurant le risque structurel empirique par exemple. C'est le parti pris par l'ensemble des méthodes de sélection. On s'intéresse pour notre part à la qualité de la distribution des étiquettes conditionnellement aux instances. Dès lors, une mesure de cette qualité doit être proposée et on adopte ici une approche descriptive.

3.1 Notations

Fixons les notations. On dispose d'un échantillon fini $D = \{X_n, Y_n\}$ de N instances étiquetées. On note $D^{(x)} = \{X_n\}$ l'ensemble des instances et $D^{(y)} = \{Y_n\}$ leurs étiquettes. Les étiquettes appartiennent à un alphabet $\mathbb{L} = \{l_j\}$ de taille J et les instances à un ensemble $\mathbb{X}$. Cet ensemble est muni d'une mesure de similitude $\delta : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_+$.

Si $P \subset \mathbb{X}$, la partition *de Voronoi* $V(P) = (V(p))_{p \in P}$ associée à P est définie par :

$$\forall p \in P,\ V(p) = \left\{ x \in \mathbb{X};\ p = \arg\min_{p' \in P} \delta(x, p') \right\}.$$

Pour $p \in P$, la *cellule de Voronoi* $V(p)$ contient les points x pour lesquels p est l'élément de P le plus similaire, relativement à δ. L'élément p est appelé *prototype* de la cellule $V(p)$. La figure 1 donnent des exemples de telles partitions.

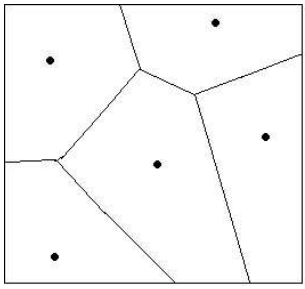

FIG. 1 – *Exemple de partitions de Voronoi.*

On définit ici un *modèle (descriptif)* comme un couple (v, ψ) où v est une partition de Voronoi formée de K cellules et ψ une matrice de taille (K, J) dont le coefficient (k, j) donne la probabilité de l'étiquette j dans la cellule k. Autrement, à chaque cellule (*i.e.* à chaque prototype) est associée une distribution de probabilité sur $\mathbb{L}$.

Si v est une partition de Voronoi composée de K cellules $v_1, \ldots, v_K$, le cardinal de $D^{(x)} \bigcap v_k$ est noté N_k ($1 \leq k \leq K$) et le cardinal de $\{X_n \in D^{(x)} \bigcap v_k;\ Y_n = l_j\}$ est noté N_{kj}. Ainsi, $N = N_1 + \cdots + N_K$ et $N_k = N_{k1} + \cdots + N_{kj}$.

3.2 Formalisation

On considère le modèle comme étant aléatoire et on cherche à spécifier la probabilité jointe $P(V, \Psi, D^{(y)}/D^{(x)})$ du modèle (V, Ψ) et des étiquettes $D^{(y)}$ connaissant les instances $D^{(x)}$. Cette probabilité est décomposée à l'aide de la formule des probabilités itérées. Du fait qu'on s'intéresse à cette probabilité, et non à la probabilité de mauvaise classification du classifieur associé, comme décrit par Vapnik (1996), on qualifie l'approche de descriptive.

Plus précisément, si K désigne le nombre de cellules de V, on commence par écrire :

$$P(V, \Psi, D^{(y)}/D^{(x)}) = P(K, V, \Psi, D^{(y)}/D^{(x)}),$$

ce qui permet de comparer des ensembles de prototypes de différentes tailles. On itère ensuite la dépendance en utilisant la formule de Bayes :

$$P(V, \Psi, D^{(y)}/D^{(x)}) = P(K/D^{(x)})P(V/K, D^{(x)})P(\Psi, D^{(y)}/K, V, D^{(x)}).$$

On suppose les comportements des distributions dans chaque cellule conditionnellement indépendants, ce qui donne :

$$P(\Psi, D^{(y)}/K, V, D^{(x)}) = \prod_{k=1}^{K} P(\Psi_k, D_k^{(y)}/V_k, D_k^{(x)}),$$

avec Ψ_k la ligne k de Ψ, V_k la k^{eme} cellule de V, $D_k^{(x)}$ les instances tombant dans V_k et $D_k^{(y)}$ leurs étiquettes. On applique une nouvelle fois la règle de Bayes et on obtient :

$$\begin{aligned} P(V, \Psi, D^{(y)}/D^{(x)}) &= P(K/D^{(x)})P(V/K, D^{(x)}) \\ &\quad \prod_{k=1}^{K} P(\Psi_k/V_k, D_k^{(x)})P(D_k^{(y)}/V_k, \Psi_k, D_k^{(x)}). \end{aligned}$$

3.3 Spécification

La probabilité $P(V, \Psi, D^{(y)}/D^{(x)})$ a été décomposée à l'aide de la formule de Bayes. On spécifie maintenant chacune des probabilités, en précisant à chaque étape le support de la probabilité concernée et en appliquant un a priori uniforme.

En ce qui concerne le nombre de cellules, *i.e.* la probabilité $P(K/D^{(x)})$, les valeurs possibles de K sont comprises entre 1 et N. L'application de l'a priori uniforme donne :

$$P(K/D^{(x)}) = \frac{1}{N}.$$

La partition de Voronoi est caractérisée uniquement par ses prototypes. Les ensembles de prototypes considérés sont les parties de $D^{(x)}$. Adopter un a priori uniforme nous conduirait à introduire le coefficient binomial $\binom{N}{K}$, puisque l'on doit choisir K prototypes parmi les N instances. Mais ce coefficient est symétrique relativement à K. Comme on préfère les valeurs faibles de K, on utilise le coefficient $\binom{N+K-1}{K-1}$, croissant avec K, proche de $\binom{N}{K}$ pour les faibles valeurs de K, nul pour $K = 1$, caractérisant ainsi plus finement notre préférence :

$$P(V/K, D^{(x)}) = \frac{1}{\binom{N+K-1}{K-1}}.$$

Dans la k^{eme} cellule, on exploite la dépendance aux données et on restreint le support des distributions possibles aux probabilités rationelles avec N_k pour dénominateur. Formellement, le support est

$$\left\{ \left(\frac{n_{k1}}{N_k}, \ldots, \frac{n_{kJ}}{N_k} \right) ; \sum_{j=1}^{J} n_{kj} = N_k \right\},$$

dont le cardinal est $\binom{N_k+J-1}{J-1}$ ($1 \leq k \leq K$). L'adoption d'un a priori uniforme donne :

$$P(\Psi_k / V_k, D_k^{(x)}) = \frac{1}{\binom{N_k+J-1}{J-1}}.$$

La partition et les fréquences des classes cibles dans chaque cellule sont à ce stade connues. Il reste à spécifier les étiquettes de chaque instance dans chaque cellule. Dans chaque cellule, le support est restreint relativement à la dépendance : pour la k^{eme} cellule ($1 \leq k \leq K$), le problème revient à placer les éléments de la cellule dans J urnes, sous la contrainte d'effectif N_{kj} dans la j^{eme} urne ($1 \leq j \leq J$). Le coefficient multinomial donne le nombre exact de ces possibilités et on obtient :

$$P(D_k^{(y)} / V_k, D_k^{(x)}) = \frac{1}{\frac{N_k!}{N_{k1}! \ldots N_{kJ}!}}.$$

Au final, en prenant l'opposé du logarithme de $P(M, D^{(y)} / D^{(x)})$, un modèle descriptif M est évalué par la formule suivante :

$$c(M) = \log N + \log \binom{N+K-1}{K-1} + \sum_{k=1}^{K} \log \binom{N_k+J-1}{J-1} + \sum_{k=1}^{K} \log \frac{N_k!}{N_{k1}! \ldots N_{kJ}!}.$$

Le premier terme correspond à la description du nombre de groupes, le second à la description des prototypes, le troisième à la description des fréquences des étiquettes dans les cellules et le dernier à la description de l'attribution des étiquettes aux instances dans les cellules.

Notons que, d'après l'approximation de Stirling ($\log x! \approx x \log x - x + O(\log x)$), le dernier terme de la formule se comporte asymptotiquement comme N fois l'entropie conditionnelle de la distribution des Y_n en connaissance de la fonction d'assignement associée à la partition :

$$\frac{1}{N} \sum_{k=1}^{K} \log \frac{N_k!}{N_{k1}! \ldots N_{kJ}!} \approx - \sum_{k=1}^{K} \sum_{j=1}^{J} \frac{N_{kj}}{N} \log \frac{N_{kj}}{N_k}.$$

4 Heuristique d'optimisation

On dispose d'un critère évaluant tout sous-ensemble de l'ensemble des instances. L'espace de recherche a pour cardinal 2^N, rendant la recherche exhaustive peu réaliste. On propose une nouvelle heuristique, encapsulant une optimisation gloutonne descendante d'un ensemble de prototypes dans une méta-heuristique de recherche à voisinage variable.

4.1 Optimisation gloutonne d'un ensemble de prototypes

L'heuristique gloutonne GLOUTON(P) s'applique à tout ensemble P de p prototypes. Chaque ensemble obtenu par suppression d'un élément de P est évalué. Parmi ces ensembles, celui minimisant le critère est déclaré vainqueur de l'étape. Ce procédé est itéré par application aux vainqueurs successifs jusqu'à l'évaluation finale d'un singleton. Le meilleur ensemble rencontré lors du parcours est renvoyé. Autrement dit, l'algorithme GLOUTON(P) s'écrit :

- $S \leftarrow P$
- **Pour** $K = p - 1$ à 1 **Faire**
  - $S \leftarrow$ le meilleur sous-ensemble de S obtenu par suppression d'un élément de S
- **Retourner** le meilleur ensemble de prototypes rencontré

Cette méthode évalue un $O(p^2)$ ensembles et chaque évaluation nécessite pour chaque instance de rechercher son plus proche prototype. GLOUTON(P) possède donc basiquement une complexité temporelle un $O(Np^3)$. Des astuces d'implantation réduisent la complexité algorithmique à un $O(Np \log p)$.

A chaque étape, toute suppression d'un prototype conduit à réattribuer uniquement les instances appartenant à la cellule de ce prototype. A l'étape K, les N instances ne sont donc à traiter qu'une fois, pour un coût qu'on peut rendre constant.

En effet, si l'on dispose pour chaque instance de la liste triée des éléments de P par distance croissante, l'acquisition du plus proche prototype suivant se fait à coût constant. L'algorithme GLOUTON(P) se voit donc adjoindre une phase d'initialisation qui devient prépondérante avec une complexité temporelle un $O(Np \log p)$ (construction de N listes triées de taille p). Le stockage de ces N listes induit une complexité spatiale en $O(Np)$.

La valeur du critère est également mise à jour à coût constant. Seuls les deux derniers termes dépendent de la répartition des instances dans les cellules. La suppression d'un prototype conduit tout d'abord à soustraire sa participation à la valeur du critère. Ensuite, la réattribution d'une instance à son plus proche prototype k suivant induit une simple incrémentation unitaire des compteurs N_k et N_{kj_0}, où j_0 est l'indice de la classe à laquelle appartient l'instance. Le terme du critère porté par le prototype k est donc mis à jour en ajoutant $\log(N_k + J) - \log(N_{kj_0} + 1)$.

Notons que l'on peut introduire dans GLOUTON une contrainte de préservation de certains prototypes : si $P' \subset P$, GLOUTON(P,P') n'évalue la suppression d'un prototype que si celui-ci n'appartient pas à P'. Dans l'optique d'une inclusion dans une méta-heuristique, cette modification permet de limiter la redondance de la recherche.

4.2 Recherche à voisinage variable

L'heuristique gloutonne est par nature susceptible de s'empêtrer dans un optimum local. Il est donc naturel d'envisager la remise en question de la solution proposée par GLOUTON. Pour cela, on applique la méta-heuristique de recherche à voisinage variable décrite par Hansen et Mladenovic (2001). Celle-ci consiste à modifier localement une solution et à réappliquer l'heuristique de base, ici l'heuristique gloutonne. Si on n'obtient pas ainsi de meilleure solution, on réitère en explorant un voisinage plus éloigné. Sinon, on réitère en considérant un

voisinage de taille minimale de la nouvelle meilleure solution. Le nombre d'étape est usuellement contrôlé par une valeur maximale spécifiée par l'utilisateur.

Une notion de voisinage d'une solution doit être définie. Pour un ensemble P_0 de p prototypes, un *voisin* est tout ensemble de prototypes $P = P_1 \bigsqcup P_2$ tel que P_1 est inclus dans P_0 et P_2 est un ensemble d'instances appartenant aux cellules de $V(P_0)$ associées aux éléments de $P_0 \setminus P_1$. Si $t \in [0, 1]$, le voisinage $\mathcal{V}_t(P_0)$ contient tous les voisins $P = P_1 \bigsqcup P_2$ de P_0 tels qu'une proportion t de prototypes dans P_0 est remplacée par une proportion t d'instances dans l'union des cellules correspondantes (cf Figure 2).

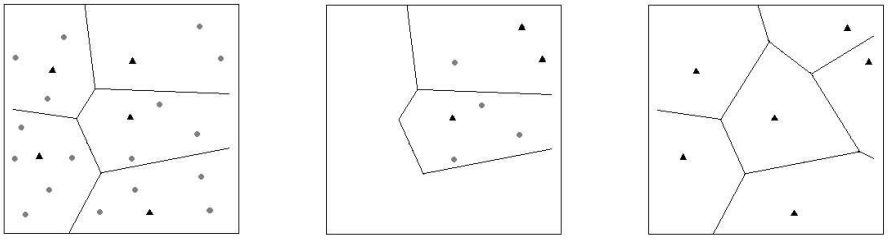

FIG. 2 – *Exemple de partitions voisines pour $t = 0.35$. (a) Répartition des instances dans les cellules de la partition. (b) 2 prototypes (soit 35% des prototypes) sont remis en cause et 3 instances (soit 35% des instances appartenant aux cellules associées) les remplacent. (c) Partition voisine obtenue.*

Un unique paramètre *Niveau* quantifie le degré d'optimisation souhaité par l'utilisateur. Une incrémentation unitaire de ce paramètre revient à doubler le temps consacré à l'optimisation. L'algorithme RVVGLOUTON($Niveau$), de complexité au pire un $O(2^{Niveau} N^2 \log N)$, s'écrit alors :

- $DegreMax \leftarrow 2^{Niveau}$
- $S_0 \leftarrow$ GLOUTON(D)
- $S \leftarrow S_0$
- $Degre \leftarrow 1$
- **TantQue** $Degre < DegreMax$ **Faire**
  - $t \leftarrow Degre/DegreMax$
  - $S' \leftarrow$ Sélection d'une solution dans $\mathcal{V}_t(S)$
  - $S \leftarrow$ GLOUTON$(S', S \bigcap S')$
  - **Si** S' est meilleur que S_0
    - $S_0 \leftarrow S'$
    - $Degre \leftarrow 1$
  - **Sinon**
    - $Degre \leftarrow Degre + 1$
- **Fin TantQue**
- **Retourner** S_0

5 Expérimentation

On évalue les méthodes de sélection d'instances selon trois axes : la performance prédictive (*i.e.* le taux de bonne classification en test), le taux de compression (*i.e.* le rapport du nombre de prototypes au nombre d'instances) et la robustesse (*i.e.* le rapport du taux de prédiction en test au taux de prédiction en apprentissage). Ces indicateurs sont estimés par validation croisée stratifiée à 10 niveaux.

	RVVG	IB3	DROP3	Explore	NN
Iris	95	92	89	95	95
Wine	83	74	71	84	83
Sonar	67	78	82	72	85
Heart	71	56	62	72	63
Bupa	68	54	57	63	61
Ionosphere	88	88	88	87	91
Australian	73	60	68	71	68
Crx	72	62	67	71	67
Breast	97	93	95	96	96
Pima	72	61	68	74	69
Vehicle	63	64	65	63	68
German	71	61	66	70	69
Led	64	67	59	58	72
Yeast	51	46	51	56	54
Segmentation	88	95	92	92	97
Abalone	25	20	23	26	21
Spam	85	77	81	82	85
Waveform	79	70	76	83	77
WaveformNoise	79	68	74	81	76
PenDigits	96	98	94	98	100
Moyenne	74.4	69.2	71.5	74.7	74.8
V/E/D		13/4/3	15/3/2	2/13/5	7/8/5

TAB. 1 – *Taux de bonne prédiction de notre méthode RVVGLOUTON(5) (RVVG), d'IB3, DROP3, Explore et de la classification par le plus proche voisin (NN), estimés par validation croisée stratifiée à 10 niveaux. Le nombre de Victoire/Egalité/Défaite significatives (au sens de la statistique de Student au niveau 0.05) de RVVG est reporté.*

Notre méthode est comparée à IB3, DROP3, Explore et la règle de classification par le plus proche voisin NN. Le niveau de RVVGLOUTON est fixé à 5, ce qui donne un temps de calcul du même ordre que celui d'Explore. Les jeux de données sont issus de l'UCI (Blake et Merz, 1996). Les jeux de données utilisés (tableau 3) sont ceux pour lesquels la performance prédictive de la règle de classification par le plus proche voisin est significativement supérieure à celle du prédicteur majoritaire (qui attribue à toute nouvelle instance la classe majoritaire sur l'échantillon). Afin d'éviter toute interférence sur les résultats relative au choix de la distance, du type de prétraitement, etc, on ne considère que des jeux de données sans valeurs

manquantes, que les variables continues. La distance de Minkowski L_1 fait office de mesure de similitude.

Le taux de bonne prédiction est reporté dans le tableau 1 et le taux de compression moyen dans le tableau 2. Les méthodes classiques, représentées par IB3 et DROP3, réalisent une compression de l'ordre de 20 à 30 pour cent, avec une perte en terme de performance prédictive (69.2% et 71.5% respectivement contre 74.8% pour la classification par le plus proche voisin). Notre méthode sélectionne un minimum d'instances (1.7% contre 2.5% pour Explore en moyenne) sans que la performance prédictive n'en soit affectée.

Les méthodes se plaçant dans le cadre de l'apprentissage de modèles sont donc plus performantes que les méthodes classiques basées sur des définitions (nécessairement heuristiques) d'utilité individuelle. L'approche descriptive adoptée ici permet de gagner encore en compression par rapport à Explore. Ceci conduit à améliorer la robustesse (tableau 2).

La fiabilité du résultat est une propriété importante, que l'approche descriptive permet encore d'améliorer. Ceci est d'autant plus intéressant que les modèles considérés (des fonctions de probabilités conditionnelles) sont plus riches que les modèles usuels (des classifieurs) : à chaque prototype est associé une distribution de probabilité sur les étiquettes. Notre méthode extrait donc plus de connaissance, et de manière plus fiable. Ceci rend profitable son utilisation en phase de préparation des données, là où les méthodes prédictives sont inadaptées.

		RVVG	IB3	DROP3	Explore	NN
Compression	Moyenne	1.7	32.0	22.2	2.5	100
	V/E/D		20/0/0	20/0/0	13/7/0	20/0/0
Robustesse	Moyenne	0.97	0.81	0.86	0.94	0.76
	V/E/D		13/7/0	12/8/0	5/15/0	16/4/0

TAB. 2 – *Compression et robustesse moyenne des méthodes testées estimées par validation croisée stratifée et nombre de Victoire/Egalité/Défaite significatives (au sens de la statistique de Student) de notre méthode.*

6 Conclusion

La classification suivant le plus proche voisin repose sur la construction d'une partition de Voronoi. Dans cet article, nous avons proposé un critère d'évaluation des partitions induites par les ensembles de prototypes inclus dans l'ensemble des instances formant l'échantillon. L'approche descriptive adoptée ayant permis de faire porter la gestion du sur-apprentissage par le critère, nous avons également proposé une heuristique d'optimisation poussée de ce critère.

Les expériences sur jeux de données de l'UCI ont montré que notre méthode est compétitive en terme de performance prédictive, tout en sélectionnant un minimum d'instances. L'étude de la robustesse a également illustré le fait que la méthode ne sur-apprend pas : la décision prise est fiable et pertinente. Cela concourt à l'emploi et au déploiement de modèles de classification par le plus proche voisin. De par la richesse de la connaissance extraite, l'utilisation de la méthode n'est pas limitée à la prédiction.

Annexe

Jeux	Taille	Variables	Classes	Prédiction majoritaire
Iris	150	4	3	0.33
Wine	178	13	3	0.40
Sonar	208	60	2	0.53
Heart	270	10	2	0.56
Bupa	345	6	2	0.58
Ionosphere	351	33	2	0.64
Australian	690	6	2	0.56
Crx	690	6	2	0.56
Breast	699	9	2	0.66
Pima	768	8	2	0.65
Vehicle	846	18	4	0.26
German	1000	24	2	0.70
Led	1000	7	10	0.11
Yeast	1484	8	10	0.31
Segmentation	2310	19	7	0.14
Abalone	4177	7	28	0.16
Spam	4307	57	2	0.65
Waveform	5000	21	3	0.34
WaveformNoise	5000	40	3	0.34
PenDigits	7494	16	10	0.10

TAB. 3 – *Caractéristiques des jeux de données utilisés.*

Références

Aha, D., D. Kibler, et M. Albert (1991). Instance-based learning algorithms. *Machine learning 6*, 37–66.

Berndt, D. et J. Clifford (1996). Finding patterns in time series : a dynamic programming approach. Technical report, Advances Knowledge Discovery Data Mining.

Blake, C. et C. Merz (1996). Uci repository of machine learning databases. http ://www.ics.uci.edu/mlearn/MLRepository.html.

Cameron-Jones, R. (1995). Instance selection by encoding length heuristic with random mutation hill climbing. In *Proceedings of the eighth australian joint conference on artificial intelligence*, pp. 99–106.

Chapman, P., J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, et R. Wirth (2000). *CRISP-DM 1.0 : step-by-step data mining guide*.

Duda, R., P. Hart, et D. Stork (2001). *Pattern classification*. New-York : Wiley & sons.

Fix, E. et J. Hodges (1951). Discriminatory analysis. nonparametric discrimination : Consistency properties. *Technical Report 4, Project Number 21-49-004, USAF School of Aviation Medicine, Randolph Field, TX*.

Gates, G. (1972). The reduced nearest neighbor rule. *IEEE transactions on information theory 18*(3), 431–433.

Hansen, P. et N. Mladenovic (2001). Variable neighborhood search : principles and applications. *European journal of operational research 130*, 449–467.

Hart, P. (1968). The condensed nearest neighbor rule. *IEEE transactions on information theory 14*, 515–516.

Sebban, M., R. Nock, et S. Lallich (2002). Stopping criterion for boosting-based data reduction techniques : from binary to multiclass problem. *Journal of machine learning research 3*, 863–885.

Stanfill, C. et D. Waltz (1986). Toward memory-based reasoning. *Communication of the ACM 29*, 1213–1228.

Vapnik, V. (1996). *The nature of statistical learning theory*. New-York : Srpinger-Verlag.

Wilson, D. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on systems, man and cybernetics 2*, 408–421.

Wilson, D. et T. Martinez (1997a). Improved heterogeneous distance functions. *Journal of artificial intelligence research 6*(1), 1–34.

Wilson, D. et T. Martinez (1997b). Instance pruning techniques. In D. Fisher (Ed.), *Proceedings of the* 14^{th} *international conference on machine learning*, San Francisco, pp. 403–411. Morgan Kaufmann.

Wilson, D. et T. Martinez (2001). Reduction techniques for instance-based learning algorithms. *Machine learning 38*(3), 257–286.

Summary

The Nearest Neighbor rule is simple and efficient. In practice, instances have to be carefully selected, in order to save computing time and to avoid overfitting. The voronoi tesselation induced by the selected instances (the prototypes) is the underlying structure of such a rule. From the descriptive approach adopted in this paper results a global evaluation criterion which makes the compromise between the number of prototypes and the discrimination of the target feature explicit. An optimisation heuristic is proposed as well. The properties of the Voronoi partitions and the criterion allows to reduce its algorithmic complexity. The method is compared with several ones from the state of the art on twenty data sets from the UCI repository. The presented technique does not suffer from any loss of the accuracy and selects less instances. As a consequence, the robustness is improved : data are not overfitted.