

## Réseaux bayésiens naïfs et arbres de décision dans les systèmes de détection d'intrusions

Nahla Ben Amor\* — Salem Benferhat\*\* — Zied Elouedi\*

\* LARODEC, Institut Supérieur de Gestion Tunis  
41 Avenue de la liberté, 2000 Le Bardo, Tunisie  
{nahla.benamorzied.elouedi}@gmx.fr

\*\* CRIL - CNRS, Université d'Artois  
Rue Jean Souvraz SP 18, 62307 Lens, cedex, France  
benferhat@cril.univ-artois.fr

---

*RÉSUMÉ. Les réseaux bayésiens sont des outils puissants pour le raisonnement et la décision sous incertitude. Une forme très simplifiée de ces réseaux est appelée réseaux bayésiens naïfs, qui disposent d'un mécanisme d'inférence particulièrement efficace. Cet article propose une étude expérimentale des réseaux bayésiens naïfs pour la détection d'intrusions. Nous montrons que ces réseaux sont satisfaisants malgré leurs structures très simplifiées. L'étude expérimentale est effectuée sur la base de données KDD'99. Trois types d'expérimentations sont réalisés en fonction du niveau de granularité des attaques. Dans toutes les expérimentations, nous comparons les performances des réseaux bayésiens naïfs à celles obtenues avec des arbres de décision, qui sont des outils très utilisés dans les problèmes de classification.*

*ABSTRACT. Bayesian networks are powerful tools for decision and reasoning under uncertainty. A very simple form of these networks is called naïve Bayes, which is particularly efficient for learning and inference tasks. This paper offers an experimental study of the use of naïve Bayes in intrusion detection. We show that eventhough they have a simple structure, naïve Bayes provide satisfactory results. We consider three levels of attack granularities depending whether we consider whole attacks, or group them into four main categories, or just focus on normal and abnormal behaviour. Different experimentations are performed in order to find the best strategies to adopt regarding the different studied cases. We compare the performance of naïve Bayes with one of well known machine learning techniques: decision trees.*

*MOTS-CLÉS : réseaux bayésiens naïfs, arbres de décision, détection d'intrusions.*

*KEYWORDS: naïve bayes, decision trees, intrusion detection.*

---

## 1. Introduction

La détection d'intrusions a pour objectif de détecter toute violation de la politique de sécurité en vigueur sur un système d'information. Elle est basée sur l'analyse à la volée ou en temps différé de ce qui se passe sur le système. On peut distinguer deux approches pour la détection d'intrusions (Axelsson, 2000) : l'approche *par scénarios* et l'approche *comportementale*. Chacune des deux approches présente des points forts ainsi que des points faibles.

L'approche par scénarios repose sur la constitution d'une base de connaissances contenant une description des attaques connues. Le module de détection d'intrusions analyse ensuite les traces d'activités journalisées à la recherche des différents événements correspondant à la description des attaques de la base de connaissances. Ces événements représentent la signature de l'attaque. Toutefois, si la signature n'est pas suffisamment précise, des faux positifs (des comportements normaux qui sont classés comme des attaques) peuvent apparaître. Par ailleurs, si la signature est trop contrainte, des variantes même très proches de l'attaque risquent de ne pas être détectées. Comme exemple de systèmes se basant sur la détection de scénarios, on peut citer les systèmes NIDES (Lunt, 1993), STAT (Ilgun *et al.*, 1995) et IDIOT (Kumar *et al.*, 1995).

Face aux difficultés de l'approche par détection de scénarios, l'approche comportementale présente certains avantages. En effet, elle repose sur l'apprentissage du comportement normal en considérant que toute déviation de ce comportement est suspecte. Donc, on n'a pas besoin de posséder une connaissance de l'attaque qui peut être menée sur le système. Plusieurs systèmes de détection d'intrusions se basant sur l'approche comportementale sont développés, on peut citer IDES (Lunt *et al.*, 1992), NIDES (Lunt, 1993) et EMERALD (Porrás *et al.*, 1997).

Récemment, Valdes et Skinner (Valdes *et al.*, 2000) ont proposé une nouvelle approche pour la détection d'intrusions basée sur les réseaux bayésiens (Jensen, 1996) (Pearl, 1988). Les réseaux bayésiens sont des outils de raisonnement avec des informations incertaines dans le cadre de la théorie des probabilités. Ils utilisent des graphes acycliques orientés pour la représentation des relations causales et des probabilités conditionnelles (de chaque nœud dans le contexte de ses parents) pour exprimer l'incertitude sur ces relations. Valdes et Skinner (Valdes *et al.*, 2000) utilisent une forme simplifiée des réseaux bayésiens, appelée *réseaux bayésiens naïfs*, composée de deux niveaux : un nœud racine qui représente la nature de la session (normal et les différents types d'attaques), et plusieurs nœuds enfants, chacun d'entre eux correspondant à un attribut la décrivant.

Les réseaux bayésiens naïfs ont plusieurs avantages dus, en particulier, à leur construction qui est très simple. L'inférence est assurée de façon linéaire (alors que l'inférence dans les réseaux bayésiens qui ont une structure générale est connue comme un problème NP complet (Cooper, 1990)). En plus, la construction des réseaux bayésiens naïfs est incrémentale, dans le sens qu'elle peut facilement être mise à jour (notamment, il est toujours possible de prendre en considération de nouvelles classes). Cependant, les réseaux bayésiens naïfs travaillent sous une hypothèse d'in-

dépendance très forte entre les attributs dans le contexte de la nature de la session. Une telle hypothèse n'est pas toujours vraie dans des applications réelles.

L'objectif de cet article est de présenter des résultats expérimentaux montrant que les réseaux bayésiens naïfs, malgré leur structure simple et leur hypothèse forte sur l'indépendance, sont satisfaisants. L'étude expérimentale est effectuée sur la base de données KDD'99 (KDD, n.d.). Les données KDD'99 sont des données formatées fournies par la DARPA à MIT Lincoln Lab. Ces données représentent 7 semaines de données libellées pour l'apprentissage et 2 semaines de données non libellées pour le test (correspondant à la simulation du trafic d'un réseau local de l'US Air force). En plus des exemples de trafic normal, les données KDD'99 contiennent les exemples de 38 attaques regroupées en quatre classes : DOS (Denial-Of-Service), R2L (Remote to User), U2R (User to Root Attacks), PROBING. Ces données sont appropriées pour évaluer un système de détection d'intrusions de type comportemental puisque la base des tests contient des attaques qui ne figurent pas dans la base d'apprentissage (il y a 14 nouvelles attaques dans la base de test).

Bien qu'il existe des travaux récents utilisant ces données (Portier *et al.*, 2000), notre étude expérimentale donne une nouvelle perspective. En particulier, nous considérons trois niveaux de granularité des attaques : le cas où l'on traite toutes les attaques, le cas où l'on regroupe les attaques en quatre classes principales et le cas où l'on se focalise uniquement sur le comportement normal ou anormal. Par ailleurs, des résultats basés sur la discrétisation des domaines continus de certains attributs sont présentés.

Afin d'évaluer les performances des résultats obtenus par les réseaux bayésiens naïfs, nous étudions et confrontons cette approche avec les arbres de décision (Mitchell, 1997) (Quinlan, 1986) (Quinlan, 1993) en utilisant les mêmes données. Nous montrons alors que la différence de performance entre les deux systèmes n'est pas très significative. Cependant, du point de vue complexité calculatoire, les réseaux bayésiens naïfs sont plus efficaces dans les phases de construction et de classification.

La section 2 présente la structure des données KDD'99. La section 3 et la section 4 présentent, respectivement, les notions de base des arbres de décision et des réseaux bayésiens qui seront illustrées par un même exemple. La section 5 décrit le plan d'expérimentation utilisé dans cet article. La section 6 considère le cas où toutes les attaques sont prises en compte en utilisant les arbres de décision et les réseaux bayésiens naïfs. Puis, la section 7 présente les résultats sur les quatre catégories d'attaques (c'est-à-dire DOS, R2L, U2R, PROBING) lorsque le regroupement est effectué avant ou après la classification. La section 8 se focalise uniquement sur le comportement normal et anormal. La section 9 donne quelques résultats sur la discrétisation des domaines continus. Enfin, la section 10 résume les principaux résultats et conclut l'article. Cet article est une version étendue de l'article (Ben Amor *et al.*, 2004).

## 2. Description des données KDD'99

L'approche par scénarios analyse des données d'audits à la recherche de scénarios d'attaques dont les signatures sont stockées dans une base de signatures. Le principe consiste à considérer que tout ce qui est décrit dans la base de signatures est intrusif, le reste est considéré comme normal.

A l'inverse, l'approche comportementale ne présuppose pas l'existence d'une base de signatures. Elle se base souvent sur un mécanisme d'apprentissage et propose de décrire le comportement « usuel » ou « normal » d'un utilisateur. Toute déviation par rapport à ce comportement normal est considérée comme une intrusion.

Les approches développées dans cet article se basent également sur des mécanismes d'apprentissage. Mais avant de les présenter, décrivons d'abord les bases de données qui sont utilisées dans nos expérimentations.

Les données utilisées dans cet article, sont celles de KDD'99 et sont orientées détection d'intrusions (KDD, n.d.). Chaque ligne code un flot de données (entre deux instants définis) entre une source (identifiée par son adresse IP) et une destination (également identifiée par son adresse IP), sous un protocole donné (TCP, UDP...). Dans la suite de l'article, nous appellerons « connexion » chaque ligne de la base KDD'99 suivant ainsi la description fournie par KDD dans (KDD, n.d.). Chaque « connexion » est caractérisée par 41 attributs tels que sa durée, le type du protocole, etc. Ces attributs ont été fixés suite à un travail de fouille de données effectués par *Lee et al.* (Lee et al., 1999). A partir des valeurs de ces attributs, chaque « connexion » dans KDD'99 est considérée comme étant une « connexion » normale ou bien une attaque. La base de données KDD'99 recense 38 attaques possibles (listées dans le tableau 1<sup>1</sup>) qui peuvent être regroupées en quatre catégories :

– **Déni de Service - « Denial-Of-Service (DOS) »** : il s'agit d'empêcher par tous les moyens les utilisateurs de se servir des ressources disponibles en temps normal. Ces attaques sont à but purement « destructeur » et sont souvent très simples à mettre en place et donnent une sensation de puissance à l'attaquant, ce qui explique leur fréquence. Un exemple d'attaques DOS est le Smurf qui provoque un déni de service via des requêtes d'écho ICMP manipulées à une adresse de diffusion d'un réseau.

– **Les attaques de type « Remote to Local access » (R2L)** : ce type d'attaque essaie d'exploiter la vulnérabilité du système afin de contrôler la machine distante. Comme exemple d'attaque R2L, il y a celle qui vise des failles des protocoles IMAP (*Internet Message Access Protocol*). Ces protocoles permettent à des utilisateurs d'accéder à leurs comptes de courrier depuis des réseaux internes ou externes.

---

1. Dans (JAM, n.d.), certaines attaques spécifiques de Httptunnel sont classées R2L, alors que d'autres attaques de type Httptunnel sont classées U2R. Dans cet article, nous les classons dans la catégorie U2R. Ces attaques n'apparaissent pas dans la base d'apprentissage, elles apparaissent uniquement dans la base de test.

– **Les attaques de type « User to Root attacks » (U2R) :** où l’attaquant essaie d’avoir les droits d’accès à partir d’un poste afin d’accéder au système. Un exemple d’attaques U2R est Rootkit, qui après avoir obtenu un accès root pour l’intrus, remplace les commandes systèmes afin qu’il puisse revenir quand il le souhaite en tant que root.

– **Reconnaissance - Probing :** ces actions ne sont pas vraiment des attaques puisqu’elles ne sont pas « destructrices » au sens où elles n’empêchent pas une entité de fonctionner correctement, mais permettent d’acquérir des informations parfois cruciales pour mener une attaque de plus grande envergure plus tard. Un exemple d’outils de reconnaissances Probing est Satan (*Security Administrator Tool for Analyzing Networks*), qui est un analyseur de ports TCP/IP qui recherche sur des hôtes distants les failles de sécurité et les défauts de configuration courants.

DOS	Probing	R2L	U2R
Apache2	Ipsweep	Ftp_write	Buffer_overflow
Back	Mscan	Guess_passwd	Httpunnel
Land	Nmap	Imap	Loadmodule
Mailbomb	PortswEEP	Multihop	Xterm
Neptune	Saint	Named	Perl
Pod	Satan	Phf	Ps
Processtable		Dict	Rootkit
Smurf		SnmPguess	
Teardrop		Spy	
Udpstorm		Sqlattack	
		WareZclient	
		WareZmaster	
		Xlock	
		Xsnoop	
		Guest	

**Tableau 1.** *Types d’attaques*

Les attributs caractérisant chaque « connexion », sont détaillés dans le tableau 2. Certains attributs sont de type discret (admettant un nombre fini de valeurs), d’autres sont de type continu.

### 3. Arbres de décision

Les arbres de décision (Quinlan, 1986) (Quinlan, 1993) représentent l’une des techniques les plus connues et les plus utilisées en classification. Leur succès est notamment dû à leur aptitude à traiter des problèmes complexes de classification. En effet, ils offrent une représentation facile à comprendre et à interpréter, ainsi qu’une capacité à produire des règles logiques de classification.

---

A1	duration	durée de la « connexion » (nb de secondes)
A2	protocol_type	type du protocole, ex. tcp, udp, icmp...
A3	service	service réseau (destination) ex. http, telnet
A4	flag	statut de la « connexion » (normal ou erreur)
A5	src_bytes	nb de données (en octets) de la source vers la destination
A6	dst_bytes	nb de données (en octets) de la destination vers la source
A7	land	1 si la « connexion » est de/vers le même hôte/port ; 0 sinon
A8	wrong_fragment	nb de fragments « erronés »
A9	urgent	nb de paquets urgents
A10	hot	nb d'indicateurs « hot »
A11	num_failed_logins	nb d'essais login ratés
A12	logged_in	1 si succès du login ; 0 sinon
A13	num_compromised	nb de conditions de « compromis »
A14	root_shell	1 si la racine shell est obtenue ; 0 sinon
A15	su_attempted	1 s'il ya tentative de la commande « racine su » ; 0 sinon
A16	num_root	nb d'accès à la « racine »
A17	num_file_creations	nb de créations d'opérations de fichiers
A18	num_shells	nb de shell prompts
A19	num_access_files	nb d'opérations sur les fichiers de contrôle d'accès
A20	num_outbound_cmds	nb de commandes outbound dans une session ftp
A21	is_hot_login	1 si le login appartient à la liste « hot » ; 0 sinon
A22	is_guest_login	1 si le login est login « invité » ; 0 sinon
A23	count	nb de connex. pour le <i>même hôte</i>
A24	srv_count	nb de connex. pour le <i>même service</i>
A25	serror_rate	% de connex. pour le <i>même hôte</i> ayant l'erreur « SYN »
A26	srv_serror_rate	% de connex. pour le <i>même service</i> ayant l'erreur « SYN »
A27	rerror_rate	% de connex. pour le <i>même hôte</i> ayant l'erreur « REJ »
A28	srv_rerror_rate	% de connex. pour le <i>même service</i> ayant l'erreur « REJ »
A29	same_srv_rate	% de connex. pour le <i>même hôte</i> utilisant le <i>même service</i>
A30	diff_srv_rate	% de connex. pour le <i>même hôte</i> utilisant <i>différents services</i>
A31	srv_diff_host_rate	% de connex. pour le <i>même service</i> utilisant <i>différents hôtes</i>
A32	dst_host_count	nb de connex. pour le <i>même hôte</i>
A33	dst_host_srv_count	nb de connex. pour le <i>même hôte</i> utilisant le <i>même service</i>
A34	dst_host_same_srv_rate	% de connex. pour le <i>même hôte</i> utilisant le <i>même service</i>
A35	dst_host_diff_srv_rate	% de connex. pour le <i>même hôte</i> utilisant <i>différents services</i>
A36	dst_host_same_src_port_rate	% de connex. pour le <i>même hôte</i> ayant le port src
A37	dst_host_srv_diff_host_rate	% de connex. pour le <i>même hôte</i> et le <i>même service</i> utilisant <i>différents hôtes</i>
A38	dst_host_serror_rate	% de connex. pour le <i>même hôte</i> ayant l'erreur « SYN »
A39	dst_host_srv_serror_rate	% de connex. pour le <i>même hôte</i> et le <i>même service</i> ayant l'erreur « SYN »
A40	dst_host_rerror_rate	% de connex. pour le <i>même hôte</i> ayant l'erreur « REJ »
A41	dst_host_srv_rerror_rate	% de connex. pour le <i>même hôte</i> et le <i>même service</i> ayant l'erreur « REJ »

---

**Tableau 2.** Liste des attributs

Un arbre de décision est composé de :

- **nœuds de décision** contenant chacun un test sur un attribut ;
- **branches** correspondant généralement à l'une des valeurs possibles de l'attribut sélectionné ;
- **feuilles** comprenant les objets qui appartiennent à la même classe.

L'utilisation des arbres de décision dans les problèmes de classification se fait en deux étapes principales :

- la construction d'un arbre de décision à partir d'une base d'apprentissage ;
- la classification ou l'inférence consistant à classer une nouvelle instance à partir de l'arbre de décision construit dans la première étape.

La construction d'un arbre de décision se base sur un ensemble d'apprentissage donné. Elle consiste à sélectionner pour un nœud de décision le test d'attribut *approprié* et puis de définir la classe relative à chaque feuille de l'arbre induit.

Plusieurs algorithmes ont été développés afin d'assurer la phase de construction. Parmi les algorithmes non-incrémentaux, nous citons ID3 et C4.5 développés par Quinlan (Quinlan, 1986) (Quinlan, 1993) qui sont probablement les plus populaires. Nous pouvons également mentionner l'algorithme CART de Breiman *et al.* (Breiman *et al.*, 1984).

Soit  $T$  un ensemble d'apprentissage contenant des objets qui peuvent appartenir à l'une des classes  $C_1, C_2, \dots, C_n$ . Si tous les objets appartiennent à la même classe, l'arbre de décision relatif à l'ensemble d'apprentissage sera une feuille libellée par cette classe, autrement  $T$  contient des objets appartenant à différentes classes. Soit  $A_k$  un attribut avec comme valeurs possibles  $(v_1, v_2, \dots, v_f)$ . L'ensemble d'apprentissage sera partitionné en sous-ensembles  $T_1, T_2, \dots, T_f$  où chaque ensemble  $T_t$  ( $t = 1, \dots, f$ ) correspond à une valeur  $v_t$  du test  $A_k$ . Puis, la même procédure sera appliquée récursivement à chaque sous-ensemble.

Signalons que l'algorithme pour la construction de l'arbre de décision à partir de l'ensemble d'apprentissage n'est pas déterministe. Ainsi, plusieurs arbres de décision peuvent engendrer la classification correcte de tous les objets de l'ensemble d'apprentissage (Marsala *et al.*, 1997).

**Exemple 1.** Afin d'illustrer les différentes notions concernant les arbres de décision et les réseaux bayésiens naïfs, nous allons considérer un exemple de base d'apprentissage donné dans le tableau 3, composé de quelques lignes de la base KDD'99. Chaque « connexion », pour des raisons de simplicité, est uniquement décrite par trois attributs discrets (au lieu des 41 attributs que contient la base) qui sont *protocol\_type*, *service* et *flag*. Les domaines de ces attributs sont :

- $D_{protocol\_type} = \{tcp, udp\}$  ;
- $D_{service} = \{http, domain\_u, auth, private, time\}$  ;
- $D_{flag} = \{SF \text{ (Fin de la synchronisation)}, REJ \text{ (Rejet)}, S0 \text{ (premier message de synchronisation)}, RSTO \text{ ( la « connexion » est établie mais la source a abandonné)}\}$ .

Nous traitons uniquement trois classes :  $C = \{Normal, DOS, Probing\}$ .

protocol_type	service	flag	C
tcp	http	SF	Normal
tcp	http	RSTO	Normal
tcp	http	REJ	Probing
tcp	time	SF	Probing
tcp	time	S0	DOS
tcp	auth	SF	Normal
tcp	auth	S0	DOS
tcp	private	SF	Normal
tcp	private	SF	Normal
tcp	private	REJ	Probing
tcp	private	RSTO	DOS
tcp	private	S0	DOS
udp	domain_u	SF	Normal
udp	private	SF	DOS
tcp	http	RSTO	Normal
tcp	private	RSTO	DOS
tcp	http	SF	Normal
udp	private	SF	DOS
udp	domain_u	SF	Normal

**Tableau 3.** Ensemble d'apprentissage

Les différents algorithmes de construction d'arbres de décision se basent généralement sur trois paramètres principaux :

- **Mesure de sélection d'attributs** qui permet de choisir l'attribut qui sera la racine de l'arbre. En outre, elle permet le choix des racines des sous-arbres de décision. En fait, la mesure de sélection d'attributs permet de classer les attributs entre eux et de choisir celui qui partitionne l'ensemble d'apprentissage de manière optimale réduisant par conséquent la taille de l'arbre.

Ainsi, une bonne mesure doit permettre de limiter la taille de l'arbre et de donner une cohérence sémantique aux nœuds qui le composent.

La mesure de sélection d'attributs est généralement basée sur la théorie de l'information. Nous citons comme mesures celles proposées par Quinlan : *le gain d'information* (Quinlan, 1986) et *le ratio de gain* (Quinlan, 1993). D'autres mesures sont utilisées comme *la mesure de Lopez De Mantaras* (Lopez De Mantaras, 1991), *le gain normalisé* (Jun *et al.*, 1997) (Pour plus de détails sur les mesures de sélection d'attributs voir (Liu *et al.*, 1994) (Mingers, 1989b)).

L'une des mesures de sélection d'attributs les plus utilisées est le *critère de gain d'information* de Quinlan (Quinlan, 1986) qui est appliqué dans ID3 et C4.5 et qui est défini comme suit :

$$\text{Gain}(T, A_k) = \text{Info}(T) - \text{Info}_{A_k}(T) \quad [1]$$

$$\text{où } \text{Info}(T) = - \sum_{i=1}^n \frac{\text{freq}(C_i, T)}{|T|} \log_2 \frac{\text{freq}(C_i, T)}{|T|} \quad [2]$$

$$\text{et } \text{Info}_{A_k}(T) = \sum_{v \in D(A_k)} \frac{|T_v^{A_k}|}{|T|} \text{Info}(T_v^{A_k}) \quad [3]$$

où  $\text{freq}(C_i, T)$  représente le nombre d'objets dans l'ensemble  $T$  appartenant à la classe  $C_i$  et  $T_v^{A_k}$  est le sous-ensemble d'objets pour qui l'attribut  $A_k$  prend la valeur  $v$ .

Ainsi, l'attribut ayant le gain d'information le plus élevé est sélectionné comme étant la racine de l'arbre (ou du sous-arbre) de décision.

Bien que le critère de gain d'information donne de bons résultats, il présente un sérieux inconvénient. En effet, il favorise les attributs ayant un nombre élevé de valeurs (Quinlan, 1993). Pour remédier à cet inconvénient, Quinlan propose une sorte de normalisation connue sous le nom de *ratio de gain* (*Gain\_ratio*) qui n'est autre que le gain d'information calibré par *Split\_Info* :

$$\text{Gain\_ratio}(T, A_k) = \frac{\text{Gain}(T, A_k)}{\text{Split\_Info}(A_k)} \quad [4]$$

où  $\text{Split\_Info}(A_k)$  est définie comme étant l'information contenue dans l'attribut  $A_k$  (Quinlan, 1993) :

$$\text{Split\_Info}(T, A_k) = - \sum_{a_k \in D(A_k)} \frac{|T_{a_k}^{A_k}|}{|T|} \log_2 \frac{|T_{a_k}^{A_k}|}{|T|} \quad [5]$$

– **Stratégie de partitionnement** qui permet de partitionner l'ensemble d'apprentissage courant en tenant compte de l'attribut test. Dans le cas d'attributs discrets, la stratégie consiste à tester toutes les valeurs possibles de l'attribut, alors que dans le cas d'attributs numériques, une étape de discrétisation est généralement nécessaires (Fayyad *et al.*, 1992) (Wehenkel, 1997).

– **Critère d'arrêt** qui permet de traiter les conditions d'arrêt du développement d'une partie d'un arbre ou d'un sous-arbre. Ce critère se déclenche généralement si tous les objets de l'ensemble d'apprentissage relatifs à un sous arbre de décision appartiennent à une seule classe. Ainsi, une partie de l'arbre vérifiant le critère d'arrêt peut donner naissance à une feuille.

Les différences entre les algorithmes développés apparaissent au niveau de ces trois paramètres et des différentes techniques d'élagage appliquées. Par ailleurs, certains algorithmes utilisent tous les éléments de l'ensemble d'apprentissage pour construire l'arbre de décision (par exemple : ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993)) alors que d'autres permettent la construction d'une manière incrémentale (ex : ID5 (Utgoff, 1988) (Utgoff, 1989), etc.)

**Exemple 2.** Continuons l'exemple précédent et calculons le ratio de gain des trois attributs afin de choisir la racine de l'arbre de décision. Calculons, tout d'abord, la valeur de  $\text{Info}(T)$  relative à l'ensemble d'apprentissage  $T$  :

$$\begin{aligned}\text{Info}(T) &= -\frac{\text{freq}(\text{Normal}, T)}{|T|} \log_2 \frac{\text{freq}(\text{Normal}, T)}{|T|} - \frac{\text{freq}(\text{DOS}, T)}{|T|} \log_2 \frac{\text{freq}(\text{DOS}, T)}{|T|} \\ &\quad - \frac{\text{freq}(\text{Probe}, T)}{|T|} \log_2 \frac{\text{freq}(\text{Probe}, T)}{|T|} \\ &= -\frac{9}{19} \log_2 \frac{9}{19} - \frac{7}{19} \log_2 \frac{7}{19} - \frac{3}{19} \log_2 \frac{3}{19} \\ &= 1.013.\end{aligned}$$

Le ratio de gain relatif à l'attribut *protocol\_type* est calculé comme suit :

$$\begin{aligned}\text{Info}_{\text{protocol\_type}}(T) &= \frac{15}{19} \text{Info}(T_{\text{tcp}}^{\text{protocol\_type}}) + \frac{4}{19} \text{Info}(T_{\text{udp}}^{\text{protocol\_type}}) \\ &= \frac{15}{19} \left( -\frac{7}{15} \log_2 \frac{7}{15} - \frac{5}{15} \log_2 \frac{5}{15} - \frac{3}{15} \log_2 \frac{3}{15} \right) + \frac{4}{19} \left( -\frac{2}{4} \log_2 \frac{2}{4} - \frac{2}{4} \log_2 \frac{2}{4} \right) \\ &= 0.97.\end{aligned}$$

$$\text{Donc Gain}(T, \text{protocol\_type}) = \text{Info}(T) - \text{Info}_{\text{protocol\_type}}(T) = 0.043.$$

$\text{Split\_Info}(T, \text{protocol\_type})$

$$\begin{aligned}&= -\frac{\text{freq}(\text{tcp}, T)}{|T|} \log_2 \frac{\text{freq}(\text{tcp}, T)}{|T|} - \frac{\text{freq}(\text{udp}, T)}{|T|} \log_2 \frac{\text{freq}(\text{udp}, T)}{|T|} \\ &= -\frac{15}{19} \log_2 \frac{15}{19} - \frac{4}{19} \log_2 \frac{4}{19} \\ &= 0.515.\end{aligned}$$

$$\text{Donc, Gain\_Ratio}(T, \text{protocol\_type}) = 0.083.$$

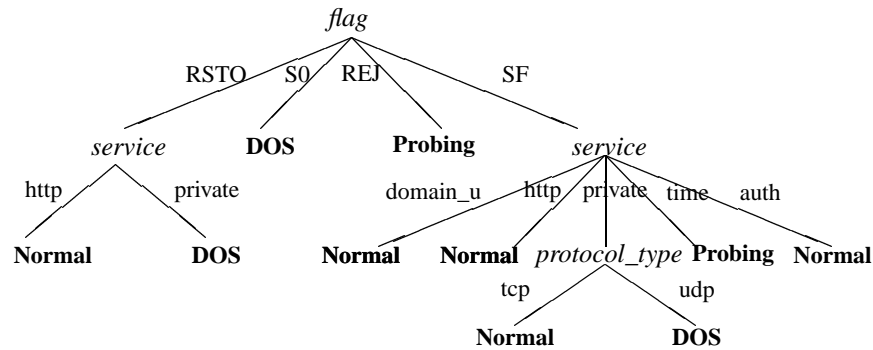
De façon similaire, nous trouvons :

$$\text{Gain\_Ratio}(T, \text{service}) = 0.25 \text{ et } \text{Gain\_Ratio}(T, \text{flag}) = 0.373.$$

Nous notons que l'attribut *flag* présente le ratio de gain le plus élevé, donc il sera choisi comme la racine de l'arbre de décision. Nous remarquons que les deux sous-ensembles d'apprentissage issus des branches portant respectivement les valeurs S0 et REJ respectent l'un des critères d'arrêt, puisque pour la valeur S0, toutes les « connexions » restantes sont des attaques DOS et pour la valeur REJ, ce sont des attaques Probing. Par conséquent les nœuds issus de ces deux branches seront déclarés comme étant des feuilles.

En suivant le même raisonnement, l'arbre final est représenté par la figure 1.

Une fois l'arbre construit, il peut être utilisé pour l'étape de classification. Pour cela, il suffit de suivre le chemin en partant de la racine jusqu'aux feuilles en effectuant les différents tests à chaque nœud selon les valeurs des attributs de l'objet à classer.



**Figure 1.** Exemple d'un arbre de décision

**Exemple 3.** Nous nous intéressons, à présent, à classer la « connexion » donnée dans le tableau 4. D'après l'arbre construit (voir figure 1), cette « connexion » va appartenir à la classe DOS. En effet, en partant de la racine (*flag*), nous suivons la branche *RSTO*, ensuite la branche *private* qui nous conduit à la classe DOS.

<b>protocol_type</b>	<b>service</b>	<b>flag</b>	<b>C</b>
udp	private	RSTO	?

**Tableau 4.** Nouvelle « connexion »

En plus des phases de construction et de classification, la plupart des algorithmes d'arbres de décision présentent une étape d'élagage (*pruning*) qui est liée à la construction de l'arbre (Mingers, 1989b) (Quinlan, 1993). Il s'agit généralement de supprimer les parties *inutiles* de l'arbre et de les remplacer par un nœud terminal qui représente la classe majoritaire des individus classés par cette partie de l'arbre. Il est à noter que d'autres critères d'élagage sont également appliqués (Mingers, 1989a). Ainsi, au lieu d'avoir un arbre de décision complet, on pourra avoir un arbre plus petit qui devrait présenter une meilleure performance de classification. Deux types d'élagage sont possibles (Esposito *et al.*, 1997) (Furnkranz, 1997) :

- le *pré-élagage* (*pre-pruning*) : appliqué au fur et à mesure que l'arbre est construit. Le pré-élagage est fortement lié aux critères d'arrêt relatifs au développement de l'arbre ;

- le *post-élagage* (*post-pruning*) : considéré comme étant l'élagage le plus courant qui est appliqué une fois la construction de l'arbre de décision est faite.

#### 4. Réseaux bayésiens naïfs

Les réseaux bayésiens (Jensen, 1996) (Pearl, 1988) sont des outils de représentation de connaissances en présence d'incertitude. Le succès de ces modèles est fortement lié à leur capacité de représenter et de manipuler des relations de (in)dépendance

qui sont importantes pour une gestion efficace des informations incertaines. Les réseaux bayésiens utilisent une représentation basée sur le conditionnement, où les connaissances sont structurées sous la forme d'un graphe acyclique orienté. Les nœuds représentent des variables et les arcs codent le lien causal (ou l'influence) entre ces variables. L'incertitude est représentée au niveau de chaque nœud en explicitant toutes les probabilités conditionnelles attachées aux valeurs associées à ce nœud sachant celles de ses parents. Cette incertitude exprime la force de la relation de *causalité* entre les variables.

Une variante simple des réseaux bayésiens est appelée réseaux bayésiens naïfs. Ces réseaux ont une structure simple et unique qui se compose de deux niveaux seulement. Le premier niveau contient un seul nœud parent et le second plusieurs enfants de ce nœud avec la forte hypothèse *naïve* d'indépendance entre les nœuds enfants dans le contexte de leur parent.

Les réseaux bayésiens naïfs sont appropriés pour le traitement des problèmes de classification (Friedman *et al.*, 1996). En effet, la classification est assurée en considérant le nœud parent comme une variable *non observée* précisant à quelle classe appartient chaque objet et les nœuds enfants comme étant des variables *observées* correspondant aux différents attributs spécifiant cet objet.

Par conséquent, en présence d'un ensemble d'apprentissage, la seule investigation à faire est de calculer les probabilités conditionnelles puisque la structure du réseau est unique. Ce calcul peut être résumé comme suit :

- les probabilités conditionnelles pour les attributs *discrets* sont calculées à partir des fréquences en comptant le nombre d'apparitions de chaque valeur d'attribut avec chacune des valeurs que le nœud parent peut prendre ;

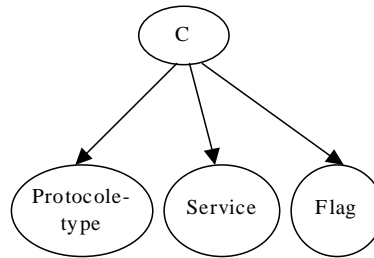
- les attributs *continus* sont généralement traités en supposant qu'ils suivent une distribution de probabilité *gaussienne* (c'est-à-dire *normale*). Donc, pour chaque valeur de classe  $c_i$  et chaque attribut continu  $A_k$ , nous devons calculer la moyenne  $\mu$  et l'écart type  $\sigma$  qui vont nous servir pour le calcul de la *fonction de densité de probabilité* pour chaque valeur  $a_k$  de  $A_k$  comme suit :

$$f(a_k) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot e^{-\frac{(a_k - \mu)^2}{2\sigma^2}} \quad [6]$$

L'hypothèse de normalité peut être considérée comme une restriction des réseaux bayésiens naïfs puisque certains attributs peuvent ne pas suivre une distribution normale. Dans ce cas, nous pouvons utiliser une méthode non-paramétrique telle que *l'estimation de densité par le noyau* qui ne suppose aucune distribution particulière pour les attributs continus (Duda *et al.*, 2000). Cette méthode est basée sur la localisation pour chaque valeur  $a_k$  d'un attribut continu  $A_K$  et les observations qui lui sont voisines à travers une fonction de pondération  $K_\sigma(a_k, a_i)$  qui affectent un poids à chaque instance  $a_i \in D_{A_K}$  basé sur sa distance par rapport à  $a_k$ . Le choix le plus commun pour  $K_\sigma$  est le noyau Gaussien  $K_\sigma = \phi(|a_i - a_k| / \sigma)$  où  $\sigma$  est l'écart type (pour un exposé plus détaillé sur l'estimation de densité par noyau voir (Duda *et al.*, 2000)). Une autre alternative serait de simplement discrétiser les attributs continus.

Dans nos expérimentations, nous avons utilisé les distributions normales et l'estimation de densité par noyau. Cependant dans la suite, nous ne présentons que l'estimation de densité par noyau (*kernel*), puisque dans toutes les expérimentations, elle a donné de meilleurs résultats. Ceci n'est pas surprenant, en effet il a été démontré dans (John, 1997) que les distributions *kernel* sont toujours meilleures que les distributions gaussiennes pour les réseaux bayésiens naïfs. Donc, dans ce qui suit, nous donnons uniquement les résultats utilisant la distribution *kernel*.

**Exemple 4.** Considérons l'ensemble d'apprentissage donné par le tableau 3, le réseau bayésien naïf correspondant à cet ensemble est représenté par la figure 2. Notons que les trois nœuds enfants correspondent aux trois attributs caractérisant les « connexions » et qu'ils sont considérés comme indépendants dans le contexte du nœud parent (*C*) qui correspond à la nature des « connexions ».



**Figure 2.** Exemple d'un réseau bayésien naïf

Le calcul des probabilités conditionnelles et *a priori* se basant sur les fréquences peut s'avérer entaché d'erreur si la valeur d'un attribut n'apparaît pas avec toutes les classes dans l'ensemble d'apprentissage. En effet, ceci peut entraîner des probabilités conditionnelles nulles qui vont réduire à zéro les probabilités de certaines classes. La technique standard pour éviter ce problème s'appelle *estimateur de Laplace* et elle consiste à ajouter 1 à tous les numérateurs et de compenser ces ajouts dans les dénominateurs. En utilisant cet estimateur, le calcul des probabilités conditionnelles et *a priori* est donné dans le tableau 5.

$P(C)$			$P(\text{protocol\_type}   C)$				$P(\text{service}   C)$				$P(\text{flag}   C)$			
N.	DOS	Prob.	N.	DOS	Prob.	N.	DOS	Prob.	N.	DOS	Prob.	N.	DOS	Prob.
10/22	8/22	4/22	tcp	8/11	6/9	4/5	http	5/14	1/12	2/8	SF	8/13	3/11	2/7
			udp	3/11	3/9	1/5	domain_u	3/14	1/12	1/8	REJ	1/13	1/11	3/7
							auth	2/14	2/12	1/8	S0	1/13	4/11	1/7
							private	3/14	6/12	2/8	RSTO	3/13	3/11	1/7
							time	1/14	2/12	2/8				

**Tableau 5.** Calcul des probabilités conditionnelles et *a priori*

Une fois le réseau quantifié, il peut être utilisé pour classer de nouveaux objets étant données leurs valeurs d'attributs en utilisant la règle de Bayes exprimée par :

$$P(c_i | A) = \frac{P(A | c_i) \cdot P(c_i)}{P(A)} \quad [7]$$

où  $c_i$  est une valeur possible de la classe  $C$  et  $A$  est l'information sur les attributs. L'information  $A$  peut être vue comme un vecteur d'instances  $a_1, a_2, \dots, a_n$  relatives aux attributs  $A_1, A_2, \dots, A_n$ , respectivement. Puisque les réseaux bayésiens naïfs travaillent sous l'hypothèse que ces attributs sont indépendants (sachant le nœud parent  $C$ ), leur probabilité conditionnelle peut être calculée comme suit :

$$P(c_i | A) = \frac{P(a_1 | c_i) \cdot P(a_2 | c_i) \cdot \dots \cdot P(a_n | c_i) \cdot P(c_i)}{P(A)} \quad [8]$$

Notons que nous n'avons pas besoin de calculer explicitement le dénominateur  $P(A)$  puisqu'il est déterminé par la condition de normalisation. Donc il est suffisant de calculer pour chaque classe  $c_i$  son degré de vraisemblance exprimé par :

$$P(a_1 | c_i) \cdot P(a_2 | c_i) \cdot \dots \cdot P(a_n | c_i) \cdot P(c_i) \quad [9]$$

afin de classer n'importe quel nouvel objet caractérisé par ses valeurs d'attributs :  $a_1, a_2, \dots, a_n$ .

Les classes choisies seront celles dont la probabilité est la plus grande.

**Exemple 5.** Supposons maintenant qu'une nouvelle « connexion » avec les valeurs données dans le tableau 4 arrive. Nous allons calculer le degré de vraisemblance de chaque valeur que la classe  $C$  peut prendre (c'est-à-dire Normal, DOS, Probing) en utilisant l'équation [9] :

- vraisemblance de Normal =  $3/11 \cdot 3/14 \cdot 3/13 \cdot 10/22 = 0.0061$  ;
- vraisemblance de DOS =  $3/9 \cdot 6/12 \cdot 3/11 \cdot 8/22 = 0.0165$  ;
- vraisemblance de Probing =  $1/5 \cdot 2/8 \cdot 1/7 \cdot 4/22 = 0.0013$ .

Ces valeurs peuvent être transformées en probabilités en les normalisant :  
 $P(Normal) = 0.2559$ ,  $P(DOS) = 0.6899$ ,  $P(Probing) = 0.0542$ .

Donc, la classe la plus probable pour la nouvelle « connexion » est DOS, qui est la même classe prédite par les arbres de décision (cf. exemple 3).

## 5. Cas d'expérimentation

La base KDD'99 contient 4 940 190 « connexions » dans l'ensemble d'apprentissage. La construction d'un arbre de décision utilisant la totalité de la base est impossible en pratique (avec un Pentium IV 2.53 Ghz, 512 M de RAM). Ce problème ne

se pose pas pour les réseaux bayésiens naïfs. Nous reviendrons plus tard sur la comparaison en terme de temps de construction entre les arbres de décision et les réseaux bayésiens.

Comme notre objectif est de comparer les deux systèmes de détection d'intrusions, nous avons uniquement traité 10 % de la base KDD'99, soit 494 019 « connexions » d'apprentissage et 311 029 « connexions » de test. La base d'apprentissage contient 19.65 % (resp. 79.07 %, 0.23 %, 0.22 %, 0.83 %) de « connexions » normales (resp. DOS, R2L, U2R, Probing) et 19.48 % (resp. 73.90 %, 5.21 %, 0.07 %, 1.34 %) de « connexions » normales (resp. DOS, R2L, U2R, Probing).

Les différentes expérimentations que nous présentons dans cet article tiennent compte des points qui suivent :

### 5.1. Nombre de niveaux de granularité d'attaques

Nous distinguons trois niveaux de granularité :

- *multiclasses* : qui traite toutes les attaques présentes dans la base KDD'99 listées dans le tableau 1, ainsi que la classe « connexion » normale ;
- *cinq-classes* : qui ne s'intéresse pas à détecter toutes les attaques, mais en plus de la classe *anormale* considère uniquement les quatre catégories d'attaques (c'est-à-dire DOS, R2L, U2R, Probing) ;
- *deux-classes* : qui se focalise uniquement sur la classe *normale* et *anormale* en groupant toutes les attaques en une même et unique classe (c'est-à-dire *anormale*).

### 5.2. Groupement des attaques

Il y a deux stratégies de regroupement des résultats, avant ou après la phase de classification :

- *groupement avant classification* : l'idée est de modifier l'ensemble des données avant de procéder à l'étape de la classification. Dans le cas des cinq classes, le regroupement se fait dans l'une des quatre catégories (DOS, R2L, U2R, ou Probing) listées dans le tableau 1. Dans le cas des deux classes, toutes les attaques sont groupées dans la classe *anormale* ;
- *groupement après classification* : nous distinguons deux cas :
  - pour le cas des cinq classes, l'ensemble d'apprentissage reste inchangé. Cependant, chaque « connexion » classée parmi les 38 attaques est assignée à l'une des quatre catégories à laquelle elle appartient ;
  - pour le cas des deux classes, il y a deux possibilités de regroupement : soit nous ne modifions pas l'ensemble d'apprentissage et chaque « connexion » classée parmi les 38 attaques est simplement libellée *anormale*, soit nous commençons par modifier l'ensemble d'apprentissage en groupant les attaques en quatre catégories, puis chaque « connexion » classée dans l'une de ces catégories sera libellée *anormale*.

### 5.3. Attributs de type pourcentage

Comme on peut le voir dans le tableau 2, certains attributs sont discrets tels que le type du protocole, alors que d'autres sont continus tels que la durée de « connexion ». Ceci n'est pas le cas de tous les attributs continus du fait que dans certains cas, le type est ambigu. Typiquement, certains attributs sont déclarés comme étant continus alors qu'ils prennent un nombre fini de valeurs.

Ceci est particulièrement vrai pour les attributs de type pourcentage. A titre d'exemple le pourcentage de « connexions » au même service ayant l'erreur « SYN » qui peut en prendre au plus 101 valeurs (le pourcentage est donné avec 2 chiffres après la virgule de 0.0 à 1.0). Donc, les variables de type pourcentage peuvent être traitées de deux façons : en tant que variables continues ou en tant que variables discrètes dont le domaine contient 101 valeurs.

L'idée est de discrétiser ces attributs afin de tester leur incidence sur les résultats de classification.

Dans la suite nous présentons les expérimentations traitant toutes les attaques (section 6), traitant les quatre catégories d'attaques (section 7), se focalisant sur le comportement normal/anormal (section 8), et montrant les apports de l'étape de discrétisation des variables de type pourcentage (section 9).

Dans chacune des expérimentations effectuées, l'évaluation de l'efficacité de classification est basée sur le *pourcentage de classification correcte* (PCC) des instances appartenant à l'ensemble test défini par :

$$PCC = \frac{\text{nombre des instances correctement classées}}{\text{nombre total des instances classées}}$$

Nous présentons aussi les matrices de confusion pour le cas cinq-classes et deux classes. La matrice de confusion donne pour chaque type de classe, la répartition des résultats de classification sur les autres classes. La diagonale de ces matrices de confusion correspond au critère « *recall* » qui donne pour chaque type de classes le taux des « connexions » correctement classées.

D'autres critères seront également utilisés, à savoir :

– *faux positifs (FP)* : pourcentage des « connexions » classées (incorrectement) comme des « connexions » normales alors qu'elles sont effectivement des attaques ;

– *faux négatifs (FN)* : pourcentage des « connexions » considérées (incorrectement) comme des attaques alors qu'elles sont effectivement des « connexions » normales.

## 6. Traitement de toutes les attaques

Cette section présente les résultats expérimentaux sur la capacité des réseaux bayésiens naïfs et des arbres de décision à détecter toutes les attaques. Ces résultats sont résumés dans le tableau 6. Les arbres de décision et les réseaux bayésiens utilisent la même base d'apprentissage et sont évalués exactement sur les mêmes données test. Le tableau 6 montre que les arbres de décision ainsi que les réseaux bayésiens naïfs sont complètement en accord avec l'ensemble d'apprentissage qui est donc cohérent. En d'autres termes, la majorité des instances d'apprentissage caractérisées par les mêmes valeurs d'attributs appartiennent à la même classe. Les deux méthodes fournissent également le même résultat pour la phase de classification avec un petit avantage pour les arbres de décision.

ENSEMBLE D' APPRENTISSAGE	ENSEMBLE TEST
<i>Arbre de décision non élagué</i>	
99.99 %	91.41 %
<i>Arbre de décision élagué</i>	
99.98 %	91.42 %
<i>Réseau bayésien naïf</i>	
99.64 %	91.13 %

**Tableau 6.** PCC relatifs au cas des multiclassés

Notons que les résultats obtenus avec les arbres de décision élagués et non élagués sont pratiquement les mêmes. La seule différence réside dans la taille de l'arbre qui contient 816 nœuds s'il est élagué et 1 228 s'il est non élagué. Pour des raisons de simplicité, dans ce qui suit nous présentons uniquement les résultats relatifs aux arbres de décision non élagués. Vu le grand nombre d'attaques, nous ne donnons pas ici la matrice de confusion.

## 7. Traitement des quatre catégories d'attaques

Cette section présente les résultats expérimentaux obtenus en classant les « connexions » dans les 5 catégories : normal, DOS, R2L, U2R, Probing. Pour cela, nous avons regroupé ensemble les attaques appartenant à la même classe. Ce traitement est effectué avant et après la phase de classification. Pour le dernier cas, nous avons utilisé les résultats relatifs à toutes les attaques en additionnant le nombre d'occurrences associées à chaque catégorie d'attaque (DOS, R2L, U2R, Probing). Le tableau 7 donne le pourcentage de classification correct (PCC) relatif à ces deux expérimentations.

Comme dans le cas de toutes les attaques, le PCC relatif aux arbres de décision dépasse légèrement celui relatif aux réseaux bayésiens naïfs dans les phases d'apprentissage et de classification.

ENSEMBLE D' APPRENTISSAGE	ENSEMBLE TEST
<i>Arbre de décision non élagué</i>	
99.99 % (99.99 %)	92.28 % (91.81 %)
<i>Réseau bayésien naïf</i>	
98.85 % (99.67 %)	90.83 % (91.40 %)

**Tableau 7.** PCC relatif aux cinq classes (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques en cinq classes après classification)

Notons que le regroupement avant et après la classification n'a aucune influence sur les arbres de décision. Cependant, avec les réseaux bayésiens naïfs, il est mieux d'effectuer le regroupement après la classification.

Le tableau 8 présente les matrices de confusion qui offrent une analyse plus détaillée du tableau 7. Ces matrices montrent que les « connexions » de type Normal, DOS et, à un degré moindre, les « connexions » de type Probing sont bien classées avec les deux techniques. Ceci n'est pas le cas avec les « connexions » de type R2L et U2R qui sont toujours mal classées.

En termes de faux positifs et faux négatifs, les arbres de décision engendrent très peu de faux positifs (presque toutes les « connexions » normales sont bien classées). En revanche, le taux de faux négatifs est peu important avec les deux approches puisque la plupart des « connexions » anormales mal classées sont déclarées comme normales.

Expliquons maintenant les raisons du mauvais classement des « connexions » U2R et R2L. Concernant les arbres de décision, signalons d'abord que ce comportement ne reflète pas les résultats optimistes obtenus avec l'ensemble d'apprentissage où 82.69 % (resp. 98.93 %) des « connexions » U2R (resp. R2L) sont bien-classées. Ensuite les proportions, dans l'ensemble d'apprentissage, des attaques U2R et R2L sont très faibles (0.22 % pour U2R et 0.23 % pour R2L).

Par ailleurs, pour les arbres de décision, lorsqu'une classe est faiblement représentée dans l'ensemble d'apprentissage, cette dernière est mal apprise, ce qui entraîne une mauvaise classification des « connexions » tests appartenant réellement à cette classe.

Finalement, le mauvais classement des « connexions » R2L et U2R est particulièrement expliqué par le fait que dans la base de test certaines valeurs d'attributs des « connexions » de type R2L (resp. U2R) n'apparaissent pas dans les « connexions » R2L (resp. U2R) de la base d'apprentissage.

Afin d'illustrer ceci, nous allons analyser la base de règles relative aux attaques U2R apprises avec l'arbre de décision :

– **R1** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ,  $A36 \leq 0.99$ ,  $A29 > 0.32$ ,  $A4 = SF$ ,  $A5 > 6$ ,  $A18 \leq 0$ ,  $A39 \leq 0.03$ ,  $A6 > 6$ ,  $A5 \leq 36530$ ,  $A17 \leq 0$ ,  $A33 \leq 4$ ,  $A3 = telnet$ ,  $A6 \leq 1342$ ,  $A1 > 20$  alors U2R

– **R2** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ,  $A36 \leq 0.99$ ,  $A29 > 0.32$ ,  $A4 = SF$ ,  $A5 > 6$ ,  $A18 \leq 0$ ,  $A39 \leq 0.03$ ,  $A6 > 6$ ,  $A5 \leq 36530$ ,  $A17 > 0$ ,  $A33 \leq 3$  alors U2R

– **R3** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ,  $A36 \leq 0.99$ ,  $A29 > 0.32$ ,  $A4 = SF$ ,  $A5 > 6$ ,  $A18 > 0$ ,  $A3 = telnet$  alors U2R

– **R4** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ,  $A36 > 0.99$ ,  $A5 \leq 33$ ,  $A5 \leq 19$ ,  $A32 \leq 1$ ,  $A2 = tcp$ ,  $A7 = 0$ ,  $A3 = ftp\_data$ ,  $A4 = SF$  alors U2R

– **R5** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ,  $A36 > 0.99$ ,  $A5 \leq 33$ ,  $A5 \leq 19$ ,  $A32 > 1$ ,  $A6 > 1$ ,  $A3 = ftp\_data$ ,  $A12 = 1$  alors U2R

– **R6** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 > 0$ ,  $A4 = SF$ ,  $A5 \leq 971$ ,  $A5 \leq 132$ ,  $A37 > 0.01$  alors U2R

– **R7** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 > 0$ ,  $A4 = SF$ ,  $A5 \leq 971$ ,  $A5 \leq 132$ ,  $A1 \leq 140$ ,  $A33 \leq 2$ ,  $A17 > 2$  alors U2R

– **R8** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 > 0.48$ ,  $A6 > 6$ ,  $A40 \leq 0.45$ ,  $A11 \leq 0$ ,  $A5 \leq 1$  alors U2R

– **R9** : si  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 > 0$ ,  $A5 \leq 10073$ ,  $A6 > 0.09$ ,  $A13 \leq 13$ ,  $A3 = telnet$ ,  $A6 \leq 6223$  alors U2R

– **R10** : si  $A23 > 64$ ,  $A6 > 1$ ,  $A33 \leq 69$ ,  $A17 > 0$  alors U2R

En analysant le sous-ensemble de la base test relatif aux attaques U2R, nous avons noté, en premier lieu, que toutes les valeurs relatives à l'attribut  $A23$ , qui est la racine de l'arbre, sont inférieures ou égales à 64 ce qui exclut l'utilisation de la règle R10. Par ailleurs, nous avons remarqué que la majorité des « connexions » tests de U2R devraient suivre les règles de R1 à R5 (187 de ces « connexions » satisfont  $A23 \leq 64$ ,  $A39 \leq 0.82$ ,  $A13 \leq 0$ ,  $A8 \leq 0$ ,  $A37 \leq 0.48$ ,  $A35 \leq 0.91$ ,  $A10 \leq 0$ ). Cependant, dans toutes ces règles, la valeur de  $A4$  doit être égale à  $SF$  ce qui n'est pas le cas dans la majorité des « connexions » tests et ceci explique leur mauvaise classification.

En effet,  $A4$  apparaît, dans les « connexions » test, avec la valeur  $REJ$  qui n'apparaît jamais dans l'ensemble d'apprentissage avec les attaques U2R (plus précisément, l'attaque Httptunnel).

La même explication est valable avec les réseaux bayésiens naïfs puisque dans la phase d'apprentissage la probabilité conditionnelle de  $REJ$  dans le contexte de U2R va être égale à zéro ( $P(REJ | U2R) \simeq 0$ ). Par conséquent, les « connexions » tests appartenant, réellement, à U2R mais ayant la valeur  $REJ$  avec l'attribut  $A4$  vont être mal classées.

<i>Arbre de décision non élagué</i>					
→	Normal	DOS	R2L	U2R	Probing
Normal (60593)	<b>99.50 %</b> ( <b>99.43 %</b> )	0.13 % (0.14 %)	0.01 % (0.02 %)	0.01 % (0.02 %)	0.36 % (0.39 %)
DOS (229853)	2.76 % (2.94 %)	<b>97.24 %</b> ( <b>96.57 %</b> )	0.00 % (0.10 %)	0.00 % (0.00 %)	0.00 % (0.39 %)
R2L (16189)	96.55 % (75.77 %)	0.02 % (2.79 %)	<b>0.52 %</b> ( <b>0.45 %</b> )	0.15 % (4.27 %)	2.76 % (16.71 %)
U2R (228)	79.82 % (23.25 %)	2.63 % (0.00 %)	1.75 % (5.26 %)	<b>7.89 %</b> ( <b>13.60 %</b> )	7.89 % (57.89 %)
Probing (4166)	19.54 % (15.22 %)	5.16 % (6.67 %)	0.34 % (0.19 %)	0.00 % (0.00 %)	<b>74.96 %</b> ( <b>77.92 %</b> )
PCC	92.28 % (91.81 %)				
<i>Réseau bayésien naïf</i>					
→	Normal	DOS	R2L	U2R	Probing
Normal (60593)	<b>96.97 %</b> ( <b>98.54 %</b> )	2.86 % (1.16 %)	0.01 % (0.02 %)	0.00 % (0.05 %)	0.17 % (0.24 %)
DOS (229853)	3.32 % (3.16 %)	<b>96.25 %</b> ( <b>96.40 %</b> )	0.02 % (0.00 %)	0.00 % (0.00 %)	0.41 % (0.44 %)
R2L (16189)	94.78 % (99.39 %)	5.16 % (0.00 %)	<b>0.02 %</b> ( <b>0.16 %</b> )	0.04 % (0.45 %)	0.00 % (0.00 %)
U2R (228)	96.49 % (95.61 %)	0.00 % (0.00 %)	0.44 % (0.00 %)	<b>1.75 %</b> ( <b>3.07 %</b> )	1.32 % (1.32 %)
Probing (4166)	30.36 % (28.78 %)	8.43 % (0.38 %)	0.79 % (0.00 %)	0.05 % (0.00 %)	<b>60.37 %</b> ( <b>70.84 %</b> )
PCC	90.83 % (91.40 %)				

**Tableau 8.** Matrices de confusion relatives aux cinq classes (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques en cinq classes après classification)

## 8. Traitement des « connexions » normales et anormales

Dans cette section, nous nous intéressons au comportement normal. Afin d'assurer cet objectif, nous avons étudié les matrices de confusion et les valeurs du PCC relatives uniquement à deux classes : normale et anormale. Nous considérons, en premier lieu, le cas où le regroupement de toutes les attaques est effectué avant la classification, puis

les cas où le regroupement est effectué après la classification en utilisant tout d'abord les résultats de toutes les attaques ensuite ceux relatifs aux cinq classes.

Les résultats sont résumés dans le tableau 9 où on remarque que les valeurs de PCC relatives aux deux classes (normale et anormale) sont très élevées.

Ces valeurs montrent aussi que, comme pour les autres expérimentations, l'écart entre les arbres de décision et les réseaux bayésiens naïfs est insignifiant puisque le PCC est pratiquement le même avec ces deux approches avec un léger avantage pour les arbres de décision. Plus précisément, la matrice de confusion présentée par le tableau 10 montre que les arbres de décision sont légèrement meilleurs que les réseaux bayésiens naïfs sauf dans le cas des « connexions » normales lorsque le regroupement est effectué avant la classification.

ENSEMBLE D'APPRENTISSAGE	ENSEMBLE TEST
<i>Arbre de décision non élagué</i>	
99.99 % (99.99 %, 99.99 %)	93.02 % (93.55 %, 92.52 %)
<i>Réseau bayésien naïf</i>	
98.75 % (99.69 %, 98.91 %)	91.45 % (91.75 %, 91.55 %)

**Tableau 9.** PCC relatifs aux « connexions » normales et anormales (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques et de ceux relatifs aux cinq classes en deux classes après classification)

<i>Arbre de décision non élagué</i>		
→	Normal	Anormal
Normal (60593)	<b>98.24 %</b> <b>(99.43 %, 98.50 %)</b>	1.76 % (0.57 %, 0.50 %)
Abnormal (250436)	8.52 % (7.87 %, 9.17 %)	<b>91.48 %</b> <b>(92.13 %, 90.83 %)</b>
PCC	93.02 % (93.55 %, 92.52 %)	
<i>Réseau bayésien naïf</i>		
→	Normal	Anormal
Normal (60593)	<b>98.48 %</b> <b>(98.54 %, 96.97 %)</b>	1.52 % (1.46 %, 3.03 %)
Anormal (250436)	10.25 % (9.89 %, 9.76 %)	<b>89.75 %</b> <b>(90.11 %, 90.24 %)</b>
PCC	91.45 % (91.75 %, 91.55 %)	

**Tableau 10.** Matrices de confusion relatives aux classes normales et anormales (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques et de ceux relatifs aux cinq classes en deux classes après classification)

MULTICLASSES	CINQ-CLASSES	NORMAL ET ANORMAL
<i>Arbre de décision non élagué</i>		
91.69 %	92.06 % (92.8 % )	93.02 % (93.55 %, 92.52 %)
<i>Réseau bayésien naïf</i>		
91.20 %	91.47 % (92.10 % )	91.45 % (92.69 %, 92.40 % )

**Tableau 11.** PCC après discrétisation (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques et de ceux relatifs aux cinq classes en deux classes après classification)

### 9. Attributs de type pourcentage

Les 15 attributs définis comme des pourcentages (A25,..., A31, A34,..., A41 dans le tableau 2) ont été traités comme des variables continues. Ces attributs peuvent être également vus comme des attributs discrets puisqu'ils peuvent prendre 101 valeurs au maximum (entre 0.00 et 1.00 avec un pas de 0.01). Par conséquent, leur discrétisation peut être directement effectuée en énumérant les valeurs possibles qu'ils peuvent prendre. Les résultats de cette expérimentation sont résumés dans le tableau 11, pour les trois cas de figures : multiclasses, cinq classes et normal/anormal. Nous avons remarqué qu'il n'y a pas d'amélioration avec les arbres de décision lorsque nous discrétisons ces attributs. C'est pour cela que nous ne donnons pas leurs matrices de confusion.

La discrétisation dans les réseaux bayésiens naïfs apporte une amélioration (bien que minime) du PCC dans presque toutes les expérimentations. Ce résultat n'est pas surprenant puisque plusieurs travaux précédents ont montré que la discrétisation améliorerait les performances des réseaux bayésiens (par exemple (Dougherty *et al.*, 1995)). Par ailleurs, une analyse approfondie de la matrice de confusion relative au cas des cinq classes, présentée par le tableau 12, montre que le PCC relatif à chacune des cinq catégories (normal, DOS, R2L, U2R, Probing) s'est amélioré et que dans certains cas tels que R2L et Probing, ces valeurs sont meilleures que celles obtenues par les arbres de décision (voir tableau 8).

### 10. Résumé des expérimentations et conclusions

Les principales conclusions sont les suivantes :

– **Interprétation des résultats** : à partir du tableau 13, nous remarquons que considérer toutes les attaques, les quatre principales attaques ou uniquement les deux classes, n'affecte pas considérablement les résultats de la classification. La différence globale ne dépasse jamais 1 % entre les différentes stratégies.

→	Normal	DOS	R2L	U2R	Probing
Normal (60593)	<b>96.64 %</b> ( <b>97.68 %</b> )	2.78 % (1.29 %)	0.23 % (0.30 %)	0.11 % (0.15 %)	0.25 % (0.58 %)
DOS (229853)	3.09 % (2.75 %)	<b>96.38 %</b> ( <b>96.65 %</b> )	0.10 % (0.01 %)	0.00 % (0.00 %)	0.43 % (0.58 %)
R2L (16189)	85.09 % (88.80 %)	4.84 % (0.01 %)	<b>7.11 %</b> ( <b>8.66 %</b> )	2.92 % (1.54 %)	0.04 % (0.99 %)
U2R (228)	54.39 % (76.32 %)	0.00 % (0.00 %)	8.33 % (3.95 %)	<b>11.84 %</b> ( <b>10.96 %</b> )	25.44 % (8.77 %)
Probing (4166)	14.19 % (10.80 %)	3.36 % (0.38 %)	3.79 % (0.38 %)	0.48 % (0.10 %)	<b>78.18 %</b> ( <b>88.33 %</b> )
PCC	91.47 % (92.10 %)				

**Tableau 12.** Matrice de confusion relative aux cinq classes après discrétisation avec les réseaux bayésiens naïfs (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques en cinq classes après classification)

MULTICLASSES	CINQ-CLASSES	NORMAL ET ANORMAL
<i>Arbre de décision non élagué</i>		
<b>91.41 %</b>	<b>92.28 %</b> (91.81 %)	93.02 % ( <b>93.55 %</b> , 92.52 %)
<i>Arbre de décision non élagué avec discrétisation</i>		
91.69 %	92.06 % (92.8 %)	93.02 % (93.55 %, 92.52 %)
<i>Réseau bayésien naïf</i>		
91.13 %	90.83 % (91.40 %)	91.45 % (91.75 %, 91.55 %)
<i>Réseau bayésien naïf avec discrétisation</i>		
<b>91.20 %</b>	91.47 % ( <b>92.10 %</b> )	91.45 % ( <b>92.69 %</b> , 92.40 %)

**Tableau 13.** Résumé des expérimentations (les valeurs entre parenthèses sont relatives au regroupement des résultats de toutes les attaques et de ceux relatifs aux cinq classes en deux classes après classification)

Pour le cas normal/anormal, ce résultat signifie que les performances ne dépendent pas d'une stratégie de regroupements particulière. Cependant du point de vue calculatoire, le choix de la stratégie peut avoir son importance. Par exemple si l'on s'intéresse uniquement au comportement normal/anormal, il est préférable de regrouper les attaques avant classification.

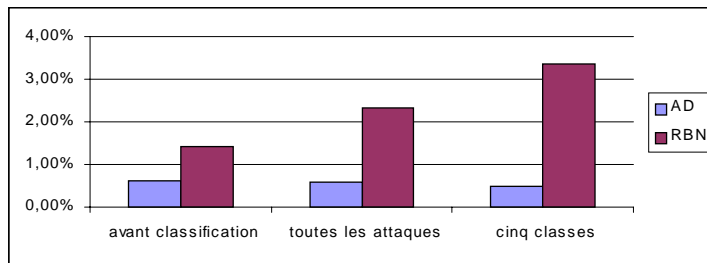
En effet, dans les réseaux bayésiens, le nombre de paramètres (probabilités conditionnelles) à stocker dans le regroupement avant classification est à peu près 20 fois plus grand que le nombre de paramètres à stocker dans le cas où le regroupement se fait après classification.

Dans le cas des quatre classes d'attaques, si la différence au niveau des PCC globaux n'est pas significative, la différence entre les PCC locaux peut être importante. Par exemple, dans le cas de la classe Probing, avec les réseaux bayésiens, la différence dépasse 10 %.

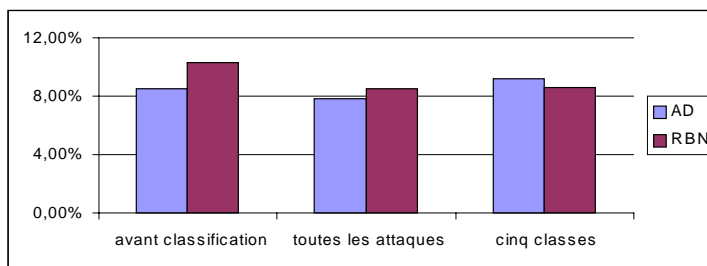
Afin d'avoir les taux des faux positifs et des faux négatifs, nous devons ramener les résultats en deux classes (normale et anormale). Pour cela, nous examinons trois cas relatifs en fonction des expérimentations réalisées :

- groupement avant classification de toutes les attaques en une seule classe ;
- groupement après classification de toutes les classes ;
- groupement après classification des quatre classes d'attaques.

Les figures 3 et 4 montrent qu'avec les deux approches le taux des faux positifs (entre 0.5 % et 3 %) est beaucoup moins important que le taux des faux négatifs (de 7.87 % à 10.25 %).



**Figure 3.** *Faux négatifs*

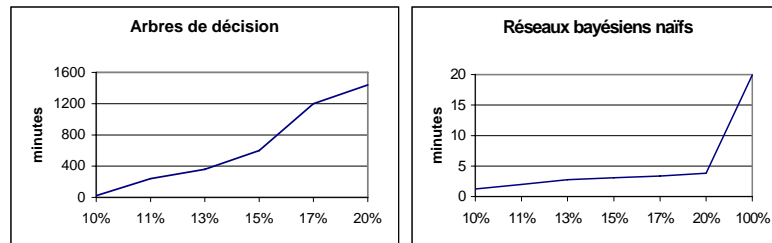


**Figure 4.** *Faux positifs*

– **Temps d'exécution** : dans le tableau 13, il est clair que les résultats donnés par les arbres de décision sont légèrement meilleurs que ceux obtenus avec les réseaux bayésiens. Cependant du point de vue complexité calculatoire, la construction d'un réseau bayésien naïf (qui se fait en un temps polynomial) est largement moins coûteuse que la construction d'un arbre de décision, où la recherche d'un arbre de décision minimal est une opération NP-complète.

Cette différence de complexité s'est vue lors des expérimentations et plus précisément sur les temps d'exécution. En effet, la figure 5<sup>2</sup> montre l'écart entre les arbres de décision et les réseaux bayésiens naïfs lors de leur construction (dans le cas de multiclassés).

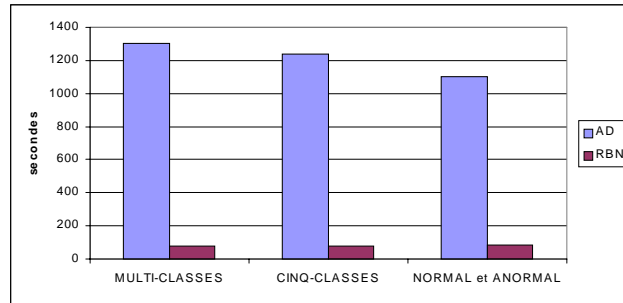
Les résultats sont sans ambiguïté : i) sur les 10 % de la base, la construction des réseaux bayésiens se fait en moyenne en une minute et est 20 fois plus rapides que la construction d'un arbre de décision, ii) l'écart devient plus important avec 11 %, 13 %, 15 %, 17 % de la base, iii) aux environs de 20 % la construction des arbres de décision dépasse une journée d'exécution et ça bloque complètement (problème de mémoire) si on essaie de traiter la totalité de la base. Les temps comparatifs pour les différents cas d'expérimentation avec 10 % de la base sont données par la figure 6.



**Figure 5.** Temps comparatif entre la construction des arbres de décision et des réseaux bayésiens naïfs

– **Travaux similaires** : le tableau 14 montre que les deux techniques présentées dans ce papier sont compétitives avec la stratégie gagnante dans KDD'99 (KDD, n.d.) qui est une variante des arbres de décision basée sur la notion de *bagging* et *boosting*. Il montre aussi que même si nous n'avons pas utilisé ces deux options dans nos expérimentations, il existe des situations où les réseaux bayésiens et les arbres de décision donnent d'aussi bons résultats, voire légèrement meilleurs que la stratégie gagnante. En effet, il existe des cas où les arbres de décision donnent de meilleurs PCC avec les classes Normal, DOS et R2L, et d'autres cas où les réseaux bayésiens naïfs sont plus performants avec les classes U2R et Probing.

2. Pour une meilleure visibilité du graphique, nous avons séparé la courbe relative aux arbres de décision et celle des réseaux bayésiens naïfs.



**Figure 6.** Temps comparatif avec 10 % de la base

	STRATÉGIE GAGNANTE	ARBRES DE DÉCISION	RÉSEaux BAYÉSIENS NAÏFS
Normal	<b>99.50 %</b>	<b>99.50 %</b>	97.68 %
Dos	97.10 %	<b>97.24 %</b>	96.65 %
R2L	8.40 %	0.52 %	<b>8.66 %</b>
U2R	13.2 %	<b>13.60 %</b>	11.84 %
Probing	83.3 %	77.92 %	<b>88.33 %</b>

**Tableau 14.** Comparaisons entre la stratégie gagnante, les arbres de décision et les réseaux bayésiens naïfs

Par ailleurs LIPPMANN *et al.* (Lippmann *et al.*, 2000) ont évalué différents systèmes de détection d'intrusions opérant directement sur les données DARPA'98 non formatées, alors que les expérimentations effectuées dans cet article concernent les données KDD'99 qui correspondent aux données DARPA'98 formatées.

Il est de ce fait difficile de comparer les résultats car d'une part, ils ne concernent pas les mêmes données, et d'autre part, ils n'utilisent pas les mêmes sous-ensembles d'apprentissage et de test. En effet, LIPPMANN *et al.* ont utilisé 300 exemples pour chacune des 38 attaques, alors que nous avons utilisé 10 % de la base KDD'99.

Si on fait abstraction des conditions d'expérimentation et on se limite uniquement aux PCC de chacune des classes d'attaques, les résultats obtenus dans nos expérimentations sont globalement meilleurs que ceux de LIPPMANN *et al.* En particulier, pour la classe d'attaque DOS, les réseaux bayésiens et les arbres de décision fournissent des taux avoisinant les 97 %, alors que dans les travaux de LIPPMANN *et al.*, le PCC est au alentour de 80 %.

Un travail futur serait d'exploiter la complémentarité entre les arbres de décision et les réseaux bayésiens naïfs afin de développer un méta-classifieur qui pourrait détecter toutes les catégories d'attaques.

## Remerciements

Ce travail rentre dans le cadre du projet RNTL (Réseau National des Technologies Logicielles) DICO (Détection d’Intrusions COopérative) et de l’ACI sécurité informatique DADDi (Dependable Anomaly Detection with Diagnosis). Nous remercions Frédéric Cuppens pour ses commentaires utiles ainsi que les rapporteurs pour leurs remarques très pertinentes.

## 11. Bibliographie

- Axelsson S., *Intrusion detection systems : a survey and taxonomy*, rapport de recherche, 2000.
- Ben Amor N., Benferhat S., Elouedi Z., « Naive bayes vs decision trees in intrusion detection systems », *Proceedings of ACM SAC 2004*, p. 420-424, 2004.
- Breiman L., Friedman J. H., Olshen R., Stone C. J., *Classification and regression trees*, Wadsworth & Brooks, 1984.
- Cooper G. F., « Computational complexity of probabilistic inference using bayesian belief networks », *Artificial Intelligence*, vol. 42, p. 393-405, 1990.
- Dougherty J., Kohavi R., Sahami M., « Forrests of fuzzy decision trees », *Proceedings of ICML'95 : supervised and unsupervised discretization of continuous features*, p. 194-202, 1995.
- Duda R. O., Hart P. E., Stork D. G., *Pattern classification*, Hardcover, 2000.
- Esposito F., Malerba D., Semeraro G., « A comparative analysis of methods for pruning decision trees », *IEEE Pattern Analysis and Machine Intelligence*, vol. 19, p. 476-491, 1997.
- Fayyad U. M., Irani K. B., « On the handling of continuous-valued attributes in decision tree generation », *Machine Learning*, vol. 8, p. 87-102, 1992.
- Friedman N., Goldszmidt M., « Building classifiers using bayesian networks », *Proceedings of the american association for artificial intelligence conference (AAAI'96)*, 1996.
- Furnkranz J., « Pruning algorithms for rule learning », *Machine Learning*, vol. 27, p. 139-171, 1997.
- Ilgun K., Kemmerer R. A., Porras P. A., « Probability propagation », *IEEE Transactions on Software Engineering*, vol. 21, n° 3, p. 181-199, 1995.
- JAM, <http://www1.cs.columbia.edu/sal/JAM/PROJECT>, Site internet, n.d.
- Jensen F. V., *Introduction to bayesian networks*, UCL Press, 1996.
- John G., Enhancements to the data mining process, Thèse de doctorat, University college, London, 1997.
- Jun B. H., Kim J., « A new criterion in selection and discretization of attributes for the generation of decision trees », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, n° 12, p. 1371-1375, 1997.
- KDD, <http://kdd.ccs.uci.edu/databases/kddcup99>, Site internet, n.d.
- Kumar S., Spafford E. H., « A software architecture to support misuse intrusion detection », *Proceedings of the national information security conference*, p. 194-204, 1995.
- Lee W., Stolfo S. J., Mok K. W., « A data mining framework for building intrusion detection models », *Proceedings of the 1999 IEEE symposium on security and privacy*, 1999.

- Lippmann R., Fried D., Grafi I., Haines J., Kendall K., Mcclung D., Webbber D., Webster S., Wyschograd D., Cunningham R., Zissman M., « Evaluating intrusion detection systems : The 1998 DARPA offline intrusion detection evaluation », *Proceedings of on DARPA information survivability conference and exposition (DISCEX'2000)*, p. 12-26, 2000.
- Liu W. Z., White A. P., « The importance of attribute selection measures in decision tree induction », *Machine Learning*, vol. 15, p. 25-41, 1994.
- Lopez De Mantaras R., « A distance-based attribute selection measure for decision tree induction », *Machine Learning*, vol. 6, p. 81-92, 1991.
- Lunt T., « Detecting intruders in computer systems », *Proceedings of the annual symposium and technical displays on physical and electronic security*, 1993.
- Lunt T., Tamaru A., Gilham F., Jagannathan R., Neumann P., Javitz H., Valdes A., Gravey T., A real-time intrusion detection expert system (IDES), rapport de recherche, Computer Science Laboratory, SRI International, Menlo Park, California, 1992.
- Marsala C., Bouchon Meunier B., « Forrests of fuzzy decision trees », *Proceedings of the international fuzzy systems association world congress (IFSA'1997)*, p. 369-374, 1997.
- Mingers J., « An empirical comparison of pruning methods for decision tree induction », *Machine Learning*, vol. 4, p. 227-243, 1989a.
- Mingers J., « An empirical comparison of selection measures for decision tree induction », *Machine Learning*, vol. 3, p. 319-342, 1989b.
- Mitchell T. M., « Decision tree learning », *Machine Learning*, 1997.
- Pearl J., *Probabilistic reasoning in intelligent systems : networks of plausible inference*, Morgan Kaufmman, 1988.
- Porras P. A., Neumann P. G., « EMERALD : Event monitoring enabling responses to anomalous live disturbances », *Proceedings of the national information systems security conference*, p. 353-365, 1997.
- Portier P., Froment-Curtil J., Data mining techniques for intrusion detection, rapport de recherche, University of Texas at Austin, 2000.
- Quinlan J. R., « Induction of decision trees », *Machine Learning*, vol. 1, p. 81-106, 1986.
- Quinlan J. R., *C4.5 : Programs for machine learning*, Morgan Kaufmann, San Mateo, California, 1993.
- Utgoff P. E., « ID5 : An incremental ID3 », *Proceedings of the international conference on machine learning*, Morgan Kaufmann, p. 107-120, 1988.
- Utgoff P. E., « Incremental induction of decision trees », *Machine Learning*, vol. 4, p. 161-186, 1989.
- Valdes A., Skinner K., « Adaptive model-based monitoring for cyber attack detection », *Proceedings of recent advances in intrusion detection (RAID'2000)*, p. 80-92, 2000.
- Wehenkel L., « Discretization of continuous attributes for supervised learning variance evaluation and variance reduction », *Proceedings of the international fuzzy systems association world congress (IFSA'1997)*, p. 381-388, 1997.

Article reçu le 16 janvier 2004  
Version révisée le 21 janvier 2005

**Nahla Ben amor** est maître assistante en Informatique de Gestion à l'Institut Supérieur de Gestion de Tunis, depuis juin 2004, et effectue ses activités de recherche au LARODEC (Laboratoire de Recherche Operationnelle, de DEcision et de Contrôle de processus). Elle a préparé sa thèse de doctorat en Gestion, option : Informatique de Gestion, à l'Institut Supérieur de Gestion de Tunis. Ses travaux de recherche se situent dans le domaine de l'intelligence artificielle et portent sur les modèles graphiques possibilistes, elle s'intéresse également aux applications des techniques de datamining à la détection d'intrusions. Elle est co-auteur de nombreux articles dans des revues et conférences internationales avec comité de lecture.

**Salem Benferhat** est Professeur à l'Université d'Artois, depuis février 2003, et effectue ses activités de recherche au CRIL (Centre de Recherche en Informatique de Lens). Auparavant, il était chargé de recherche CNRS à l'IRIT (Toulouse). Il a préparé son mémoire d'habilitation à diriger les recherches en informatique, sa thèse de doctorat à l'Université Paul Sabatier, Toulouse III. Ses travaux de recherche se situent dans le domaine de l'intelligence artificielle, et portent sur le raisonnement plausible, réseaux bayésiens, la fusion d'informations incertaines et la dynamique des croyances en utilisant des modèles qualitatifs. Il s'intéresse aussi aux applications de l'intelligence artificielle à la sécurité informatique. Il est membre de comités de programmes de plusieurs conférences, et est auteur ou co-auteur de nombreux articles dans des revues et conférences internationales avec comité de lecture.

**Zied Elouedi** est maître assistant en Informatique de Gestion à l'Institut Supérieur de Gestion de Tunis, depuis juin 2003, et effectue ses activités de recherche au LARODEC (Laboratoire de Recherche Operationnelle, de DEcision et de Contrôle de processus). Il a préparé sa thèse de doctorat en Gestion, option : Informatique de Gestion, à l'Institut Supérieur de Gestion de Tunis. Ses travaux de recherche se situent dans le domaine de l'intelligence artificielle et portent sur les techniques de classification dans un cadre incertain, il s'intéresse également aux applications des techniques de datamining à la détection d'intrusions. Il est co-auteur de nombreux articles dans des revues et conférences internationales avec comité de lecture.

**ANNEXE POUR LE SERVICE FABRICATION**  
A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER  
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER  
LE FICHER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :  
*RSTI-TSI. Volume 25 – n°2/2006*
2. AUTEURS :  
*Nahla Ben Amor\** — *Salem Benferhat\*\** — *Zied Elouedi\**
3. TITRE DE L'ARTICLE :  
*Réseaux bayésiens naïfs et arbres  
de décision dans les systèmes  
de détection d'intrusions*
4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :  
*RBN et AD dans les IDS*
5. DATE DE CETTE VERSION :  
*30 mars 2006*
6. COORDONNÉES DES AUTEURS :
  - adresse postale :
    - \* LARODEC, Institut Supérieur de Gestion Tunis  
41 Avenue de la liberté, 2000 Le Bardo, Tunisie  
{nahla.benamor,zied.elouedi}@gmx.fr
    - \*\* CRIL - CNRS, Université d'Artois  
Rue Jean Souvraz SP 18, 62307 Lens, cedex, France  
benferhat@cril.univ-artois.fr
  - téléphone : RBN et AD dans les IDS
  - télécopie : +33.3.21.79.17.79
  - e-mail : +33.3.21.79.17.70
7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :  
L<sup>A</sup>T<sub>E</sub>X, avec le fichier de style `article-hermes2.cls`,  
version 1.23 du 17/11/2005.
8. FORMULAIRE DE COPYRIGHT :  
Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :  
<http://www.revuesonline.com>

SERVICE ÉDITORIAL – HERMES-LAVOISIER  
14 rue de Provigny, F-94236 Cachan cedex  
Tél. : 01-47-40-67-67  
E-mail : [revues@lavoisier.fr](mailto:revues@lavoisier.fr)  
Serveur web : <http://www.revuesonline.com>