Titre: Title:	Sélection de question et choix de classificateur pour questionnaires adaptatifs
Auteur: Author:	Alexandre Spaeth
Date:	2009
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Spaeth, A. (2009). Sélection de question et choix de classificateur pour questionnaires adaptatifs [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/141/

UNIVERSITÉ DE MONTRÉAL

SÉLECTION DE QUESTION ET CHOIX DE CLASSIFICATEUR POUR QUESTIONNAIRES ADAPTATIFS

ALEXANDRE SPAETH
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU DIPLÔME DE
MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
AOÛT 2009

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

SÉLECTION DE QUESTION ET CHOIX DE CLASSIFICATEUR POUR QUESTIONNAIRES ADAPTATIFS

présenté par : M. SPAETH Alexandre.

en vue de l'obtention du diplôme de : Maîtrise ès Sciences Appliquées

a été dûment accepté par le jury constitué de :

M. GUÉHÉNEUC Yann-Gaël, Doct., président.

M. GAGNON Michel, Ph.D., membre et directeur de recherche.

M. DESMARAIS Michel, Ph.D., membre et codirecteur de recherche.

Mme. LOYE Nathalie, Ph.D., membre.

Remerciements

Je tiens à remercier en premier lieu mes directeur et co-directeur de recherche, Michel Gagnon et Michel C. Desmarais pour les conseils, les encouragements et toute l'aide qu'ils ont pu m'apporter durant ce projet de recherche.

Je remercie aussi mes parents et ma famille pour leur soutien sans faille durant toutes mes études et plus particulièrement durant les premiers mois de ma maîtrise à Montréal.

Enfin, un grand merci à tous ceux qui, des deux côtés de l'Atlantique, m'ont soutenu, aidé ou encouragé au cours des dernières années.

Résumé

Les environnements d'apprentissage prennent une place de plus en plus grande dans les techniques modernes d'éducation. L'évaluation des connaissances d'un individu est au coeur de ces environnements. Pour déterminer efficacement ce niveau de connaissances, le recours aux systèmes de test assistés par ordinateurs est de plus en plus fréquent. Notamment, on cherche généralement à minimiser la longueur des questionnaires et donc le nombre de questions posées.

Ces systèmes nécessitent la création d'un modèle de connaissance qui peut être créé par des experts du domaine ou par une approche statistique tirant parti de données existantes. Parmi ces approches statistiques, les réseaux bayésiens restent très largement utilisés. Cependant, certains auteurs démontrent qu'il existe des techniques qui semblent plus pertinentes et on observe par ailleurs que des techniques basiques de la fouille de données ne sont pas utilisées dans ce cadre.

Durant ce projet, nous cherchons à déterminer quel classificateur est le plus pertinent dans le cadre des questionnaires adaptatifs, c'est à dire dont les questions posées dépendent des réponses aux questions passées. Notamment, nous nous intéresserons plus particulièrement au cas d'une banque de données comportant un nombre important de questions, un grand nombre de réponses de répondants de sessions précédentes et pour lequel on dispose d'informations supplémentaires (thèmes des questions en particulier). Pour cela, nous comparons une méthodes sans inférence statistique, un arbre de décisions, des méthodes bayésiennes (naïve et réseau bayésien) et les réseaux *POKS* (*Partial Order Knowledge Structure*). Nous évaluons les performances de prédictions de chaque classificateur, c'est à dire surtout le taux de bonnes classifications, en gardant à l'esprit les performances en terme de temps de calcul. Cette comparaison

se base sur quatre jeux de données aux caractéristiques (nombre de questions et de répondants) différentes : un jeu comportant 34 questions et 48 répondants, un jeu composé de 160 questions et 41 répondants, un jeu comportant 20 questions et 149 répondants et enfin un jeu composé de 60 questions et 453 répondants, et issus d'articles précédents ou des archives du test de mathématiques de l'École Polytechnique de Montréal.

Les résultats montrent que le classificateur bayésien naïf est relativement très performant, en considérant sa simplicité et sa vitesse d'exécution. Les réseaux *POKS* permettent aussi d'obtenir de bonnes performances de prédiction. Ces deux classificateurs sont les plus performants avec un grand nombre de questions mais ils permettent surtout d'atteindre un taux de classifications correctes élevé avec un nombre réduit de questions posées et permettent donc dans le cadre d'un questionnaire adaptatif de réduire significativement le nombre de questions à poser. Cependant ces résultats ne se transposent pas aux données issues de la banque de données qui nous intéressait le plus et il faudrait éventuellement vérifier qu'en récoltant de nouvelles données, ce résultat est toujours valable.

Abstract

Adaptive learning environments are based on the modeling and the evaluation of a student. There are many approaches to handle this problematic. For instance, probabilistic and learning techniques are convenient because they allow to automatize the process and to describe the student skill with uncertainty.

In previous articles, authors claimed that a solution with a Bayesian framework called *POKS* (for Partial Order Knowledge Structure) can outperform Bayes network for this type of work. But the comparison between this techniques and classical data-mining techniques doesn't exist.

We compare the *POKS* approach with other data-mining techniques: zero-rule classifier, decision-tree and other Bayesian techniques. The comparison relies on two points: performance, which is the rate of good classification, and classification time. We used four different data sets with different features (34 questions vs 160 vs 20 vs 60, 48 students vs 41 vs 149 vs 453).

The simulations show that a simple method like naive Bayes can equal a more complex structure like *POKS* and both perform better than classical Bayes network or other data-mining techniques.

Table des matières

Liste des tableaux

Liste des figures

Liste des annexes

Chapitre 1

Introduction

1.1 Cadre de la recherche

Avec l'arrivée des ordinateurs multimédia et la diffusion de l'Internet, on a vu apparaître de nombreuses solutions d'apprentissage assisté par ordinateur (ou e-learning) qui permettent aux professeurs de partager des suppléments de cours, qui facilitent l'organisation du contenu pédagogique, et qui peuvent même se substituer à un cours en classe. Pour arriver à un suivi intéressant pour chaque élève, il faut s'assurer de remplir trois critères distincts qui répondent à trois problèmes bien définis. Le premier est la représentation du domaine d'apprentissage. Cela signifie être capable d'identifier les difficultés et les connaissances importantes apportées par le cours et plus généralement par le domaine étudié. Deuxièmement, afin d'avoir un environnement adapté à chaque élève, il faut être capable d'évaluer les capacités de chacun avec précision et le plus rapidement possible. En effet, pour pouvoir aider l'élève efficacement, il faut évaluer ses connaissances et ses lacunes de façon détaillée. Enfin, le troisième problème à considérer est celui du guidage qui consiste à proposer, en fonction de la connaissance initiale et du but envisagé, les leçons et les exercices appropriées et sélectionner les tests à faire passer pour vérifier que la connaissance est acquise.

Alors que le premier problème est encore bien souvent résolu à la main par des experts du domaine (même s'il existe des techniques d'automatisation), les deux derniers problèmes peuvent être plus aisément effectués à l'aide d'un ordinateur.

Une des techniques utilisées pour effectuer le diagnostic des connaissances d'un répondant est le questionnaire adaptatif, c'est à dire l'adaptation des questions posées au niveau supposé du répondant. Par exemple, les tests bien connus suivants utilisent une telle technique : les tests de certification Cisco et Microsoft ou encore les tests de niveau d'anglais TOEFL.

C'est précisément le cadre dans lequel travaille un groupe d'étudiants et de professeurs de l'École Polytechnique de Montréal Desmarais *et al.* (2006a). Ils cherchent en effet à mettre en place un environnement d'apprentissage adapté à chaque élève. Pour cela, un groupe d'experts a été chargé de créer une ontologie des connaissances mathématiques attendues d'un élève de première année au baccalauréat. Cette ontologie permettra par la suite d'effectuer une modélisation précise de l'apprenant. Le projet d'environnement d'apprentissage s'attaque à présent au second problème, c'est-à-dire la représentation et l'évaluation de la connaissance des apprenants.

Jusqu'à présent, dans le cadre de leur évaluation, il était demandé aux élèves de remplir un questionnaire à choix multiples de 60 questions qui permettait de les évaluer. On se rend compte rapidement qu'une simple informatisation de ce questionnaire apporterait peu. Il faut plutôt viser à personnaliser le questionnaire pour chaque répondant afin que celui-ci soit adapté à son niveau. C'est-à-dire que l'on ne veut pas poser des questions dont on sait qu'il connaît la bonne réponse, ni des questions pour lesquelles on sait qu'il va se tromper afin de gagner du temps en ne posant que des questions utiles. Il y a alors deux problèmes à résoudre : le choix de la prochaine question à poser et l'évaluation des réponses probables de l'élève aux questions qui n'ont pas encore été posées. Pour cela, l'utilisation des classificateurs utilisés en fouille de données devrait permettre d'obtenir de bons résultats.

La recherche de techniques de modélisation efficaces pour un répondant fait partie d'un champ de recherche très actif depuis les années 1950 qui se nomme la psycho-

métrie. On se référera par exemple à Thurstone (1959) pour avoir des exemples de travaux dans ce domaine. Cependant, les techniques bien connues d'apprentissages machines ne sont pas appliquées à ce problème déjà très étudié.

Notre projet consiste donc à déterminer, parmi une large sélection de classificateurs classiques en apprentissage machine, ceux qui sont les plus efficaces pour la modélisation des répondants. Nous désirons aussi obtenir un classificateur le moins exigent possible en temps de calcul.

1.2 Objectifs de la recherche

Le principe d'un test adaptatif est de déterminer, sans poser l'ensemble des questions existantes, si le répondant maîtrise ou non les connaissances que l'on veut évaluer. Pour cela, on utilise un classificateur qui, pour chaque question qui n'a pas encore été posée, construit un modèle basé sur les réponses données par d'autres répondants et qui, à partir des réponses fournies par le répondant aux questions qui lui ont déjà été posées, détermine les probabilités d'une bonne réponse. Si la probabilité d'échec est supérieure à la probabilité de réussite (soit $p > 0.5$), on considère que l'élève va échouer à cette question et inversement. Afin d'être le plus sûr possible de notre prévision, il faut sélectionner le meilleur classificateur existant. Un bon classificateur permet de poser moins de questions, puisque les questions ayant une forte probabilité de succès ou d'échec ne seront pas posées. Il permet aussi d'évaluer l'état des connaissances du répondant avec une faible marge d'erreur.

Notre objectif général est donc de déterminer quelle technique de classification est la plus efficace pour caractériser les connaissances d'un individu à partir d'une série de questions dans un domaine considéré. Nous visons à combler une lacune car il n'existe pas de comparatifs dans le cas des questionnaires adaptatifs.

Pour cela, nous avons sélectionné une liste de classificateurs en fonction de leurs qualités connues et nous avons comparé leur fonctionnement théorique. Par la suite, nous avons réalisé une série d'expériences pratiques afin de déterminer lequel est le plus performant sur les quatre jeux de données dont on dispose, dont notamment un jeu de données contenant un grand nombre de questions (près d'un millier) dans le domaine des mathématiques et contenant un grand nombre de réponses des élèves (archives des questionnaires papier depuis 2000).

1.3 Organisation du mémoire

Ce mémoire est découpé en 8 chapitres. Dans le prochain chapitre, on effectue une revue de littérature qui nous permet de faire ressortir l'originalité de notre travail, d'effectuer un survol des différentes techniques de l'apprentissage assisté par ordinateur et de sélectionner les outils nécessaires à la réalisation de nos travaux, tels que les différents classificateurs existants. Dans le chapitre suivant, nous revenons plus précisément sur le cadre de la maîtrise, sur les objectifs poursuivis et sur les travaux effectués. Le quatrième chapitre porte sur la description de l'ensemble des classificateurs sélectionnés accompagnée d'exemples explicatifs. Dans le cinquième chapitre nous exposerons les différentes expériences qui ont été effectuées. Les résultats sont exposés et discutés dans le chapitre suivant. Le chapitre 7 expose les travaux effectués pour mettre en place une première architecture de test en conditions réels. Et enfin, le dernier chapitre conclut les travaux et propose des pistes d'améliorations et de travaux futurs.

Chapitre 2

Apprentissage assisté par ordinateur

2.1 Justification des questionnaires adaptatifs

On peut s'interroger sur l'utilité des questionnaires adaptatifs. En effet, ces derniers permettent-ils réellement, à travers un guide d'apprentissage personnalisé en fonction des connaissances, une réelle amélioration du niveau de connaissance des élèves ? Plusieurs études démontrent que l'utilisation des questionnaires adaptatifs est réellement bénéfique. On se référera notamment à Lewis et Sheehan (1990), qui nous permet de vérifier que la durée des examens peut réellement être réduite significativement, à VanLehn *et al.* (2005), qui porte un regard sur les résultats obtenus par leur système d'apprentissage personnalisé 5 ans après sa mise en place, et à Koedinger *et al.* (1997), qui évalue un système de tutoriel intelligent utilisé dans une école et indique que les élèves à qui la matière à été enseignée via un système intelligent, c'est à dire qui s'adapte à l'élève, ont obtenus de meilleurs résultats que ceux qui ont appris avec un système fixe. On voit donc que les systèmes automatiques fonctionnent et obtiennent de bons résultats.

2.2 Construction d'un guide d'apprentissage

Afin d'avoir un système complet, il faut d'abord définir clairement le sujet que l'on cherche à enseigner (voir Aroyo et Dicheva (2004)). La définition claire d'un domaine

est généralement faite par des experts du domaine qui doivent construire eux-même les relations entre les sous-domaines et s'assurer que la définition est complète et cohérente. Cependant, de plus en plus de travaux visent soit une automatisation partielle (voir Cederberg et Widdows (2003); Cimiano (2006)), soit une automatisation complète (voir Zouaq et Nkambou (2009)) de la création d'ontologies, basée sur une étude de texte et des liens existants entre eux dans un corpus de texte rattaché au domaine, qui permettent de représenter efficacement un domaine.

Une fois la définition du domaine achevée, il faut acquérir une modélisation précise de l'apprenant. On cherche à savoir ce que l'élève sait et non la quantité de matière qu'il connaît (voir Martin et Vanlehn (1995); VanLehn et Martin (1998)). Pour cela, on peut soit utiliser des examens corrigés par des professeurs, soit, dans le cas d'un système entièrement automatique, des questionnaires à choix multiples facilement corrigibles par un ordinateur. Il faut par la suite créer un grand nombre de questions, les classer par connaissances et par compétences en fonction de la définition du domaine effectuée précédemment. Le nombre de questions nécessaires et le nombre de répondants préalables afin d'avoir un système performant a été notamment étudié dans Desmarais *et al.* (2008). Une fois ce nombre atteint, on peut créer le questionnaire adaptatif en se basant sur les données récoltées (voir Jameson (1996) par exemple). Le choix des questions à poser est documenté dans Rudner (2002).

Pour finir, il s'agit de créer un guide d'apprentissage qui se base sur l'ensemble des modèles connus (le domaine et l'étudiant) et qui propose des exercices et des tests appropriés afin de progresser dans les domaines non maîtrisés. Par exemple, on pourra consulter Brusilovsky *et al.* (1998) ou Brusilovsky *et al.* (1996) pour des exemples de guide d'apprentissage tirant parti des informations collectées jusque-là.

Quelques études ont porté sur le temps nécessaire à la création d'un tel guide d'apprentissage, afin de s'assurer que la durée consacrée à sa création ne soit pas

exagérée par rapport aux bénéfices escomptés (voir Woolf et Cunningham (1987)). Elles révèlent que cela est compliqué et justifie les efforts pour la recherche de méthodes d'automatisation du processus. De nombreuses ébauches semi-automatiques existent, par exemple Ainsworth et Fleming (2005) ou encore Koedinger *et al.* (2004) et Nuzzo-Jones *et al.* (2005), et des guides d'apprentissage automatiques commencent à voir le jour, tels que Turner *et al.* (2005).

2.3 Psychométrie et techniques de modélisation existantes

Il faut noter ici que la modélisation précise d'un individu, qui consiste essentiellement à mesurer les connaissances d'un apprenant s'appelle la psychométrie et il s'agit d'un domaine largement étudié de la psychologie. On étudiera par exemple les documents Anastasi (1994); Allen et Yen (2001); D. (2005) pour prendre connaissance de ce domaine. Une des techniques les plus étudiées par les experts de ce domaine étant la théorie de réponse à l'item (ou Item Response Theory en anglais, IRT) Hambleton *et al.* (1991, 2001) qui consiste à modéliser chaque question par une fonction de probabilité de réussite dépendant du niveau du répondant et de paramètres (au nombre de 3 en général), permettant de définir la difficulté de la question, son pouvoir discriminant et de prendre en compte la chance dans le cas d'un questionnaire à choix multiples. Pour finir sur les techniques classiques, précisons que le choix des questions à poser utilise bien souvent des techniques de minimisation de l'entropie Hald (1999) ou de maximisation de l'information (type information de Fisher Savage (1976)). Le but poursuivi par notre travail est à rapprocher de la TRI dans la mesure où nous poursuivons le même but : représenter efficacement les connaissances d'un individu.

2.4 Classificateurs

Afin de s'occuper de la modélisation de l'étudiant, les chercheurs préfèrent souvent utiliser des techniques à base de réseaux bayésiens (comme VanLehn et Niu (2001); Vomlel (2004); Heckerman (1995)) qui ont fait leur preuve sous certaines conditions d'indépendance (voir Mislevy et Chang (2000)). D'autre part, les classificateurs bayésiens naïfs peuvent donner de bons résultats (voir Rish (2005)) mais les théories d'espace de connaissances telles que développées dans Doignon et Falmagne (1999), où l'on considère que l'apprentissage des connaissances suit un sens logique, ont poussé à chercher des modèles en réseaux tels que *POKS* (voir Desmarais *et al.* (2006a)).

Dans le même ordre d'idée, on peut penser à des classificateurs tels que les arbres de décision (présentés entre autres dans Quinlan (1993)), qui représentent une certaine hiérarchie des questions. L'ensemble des techniques utilisées en fouille de données et d'apprentissage machine peuvent d'ailleurs potentiellement être utilisées. Une présentation détaillée de plusieurs de ces techniques est fournie dans Witten et Frank (2005). Le test de l'utilisation de techniques génériques de fouille de données ne semble en tout cas pas répandu et nous a donc poussé à effectuer ce travail.

Chapitre 3

Définition et cadre du projet

Dans ce chapitre, nous présentons le but principal du projet, les objectifs à atteindre ainsi que les besoins du projet. Nous expliquons aussi pourquoi il est intéressant d'utiliser un système adaptatif. Puis nous présentons le travail qui a été effectué et enfin nous donnons un aperçu de l'application client/serveur qui a été conçue.

3.1 Présentation de l'existant

Ce projet vise à fournir à l'ensemble des étudiants de baccalauréat un environnement d'apprentissage efficace et autonome, en particulier dans le domaine des mathématiques.

Jusqu'à présent, les élèves pouvaient, volontairement, passer un test d'évaluation afin de cerner leurs lacunes. Ce questionnaire à choix multiples, divisé en 60 questions, permettait d'évaluer les élèves suivant 7 thèmes : notions élémentaires, géométrie, calcul différentiel, calcul intégral, calcul matriciel, trigonométrie et représentation vectorielle.

Ce test doit être fait en classe dans une durée de deux heures. La correction est assurée par des professeurs de Polytechnique et les résultats sont ensuite communiqués aux élèves après correction. Cela nous donne déjà plusieurs contraintes, à savoir un temps de correction réduit, la communication des résultats dès la fin du test et une réduction de la durée du test. Il faudrait aussi pouvoir l'effectuer depuis n'importe

quel ordinateur connecté à l'Internet. Enfin, il devrait pouvoir être effectué à volonté et à n'importe quel moment de l'année.

3.2 Objectifs et besoins du projet

Une vision globale de la situation actuelle permet d'identifier facilement des objectifs simples, par exemple informatiser le test, et des objectifs plus complexes, notamment réduire intelligemment la durée de l'examen.

Informatiser un questionnaire à choix multiple est un objectif atteint par de nombreux systèmes d'apprentissages. Idéalement, il faudrait donc que la solution choisie puisse facilement être portée sur la plate-forme d'apprentissage en ligne utilisée à l'École Polytechnique, à savoir Moodle.

La seconde étape consiste à réduire efficacement la durée de l'examen, ou du moins pouvoir l'adapter à la disponibilité du répondant. Pour cela, outre le fait que la session en cours devra pouvoir être sauvegardée localement, et donc que le serveur ne devra pas avoir à sauvegarder quoi que ce soit, il faut adapter le questionnaire en fonction du répondant. Cela paraîtrait en effet naturel à un interrogateur d'interrompre le test une fois les connaissances de l'individu cernée. De la même façon, le questionnaire automatique devrait s'adapter au répondant pour s'arrêter dès que la connaissance est suffisante, mais aussi pour accélérer la modélisation de l'étudiant, par exemple en adaptant le niveau des questions qui sont posées.

Il faut donc remplir deux conditions afin d'obtenir la meilleure évaluation des répondants possible. La première est d'avoir un mécanisme d'évaluation pertinent et efficace et la seconde est de pouvoir déterminer le plus efficacement possible quelle est la prochaine question à poser pour avoir un questionnaire efficace.

On a donc identifié deux objectifs prioritaires : tout d'abord identifier une méthode

d'évaluation du répondant qui puisse être mesurée en termes de précision. Puis mettre en place une méthode de détermination de l'ordre des questions à poser.

Cela rejoint l'objectif scientifique de notre travail : effectuer un comparatif des classificateurs existants dans le cadre des questionnaires adaptatifs.

Le questionnaire de base étant sous-divisé en thème, et l'objectif étant même de représenter le champ des connaissances grâce à une ontologie, on peut se baser sur cette classification pour essayer d'atteindre l'un ou l'autre des objectifs.

Les besoins auxquels il faut répondre sont à présent identifiables : tout d'abord, il faut dresser une liste des classificateurs susceptibles de fournir de bons résultats lors de la représentation des élèves ou du choix des questions ; ensuite, il faut récupérer les résultats des années précédentes et éventuellement les pré-traiter afin de pouvoir entraîner et tester les différents classificateurs que l'on aura sélectionné. Pour vérification, on se procurera aussi d'autres questionnaires qui ont déjà fait leur preuves dans ce domaine pour s'assurer de la cohérence des résultats que l'on obtiendra avec les questionnaires de mathématiques.

3.3 Intérêt du système adaptatif

Contrairement à un questionnaire à choix multiples statique, un système adaptatif permet, avec un même nombre de questions, d'avoir une meilleure évaluation de chaque répondant et donc d'avoir un aperçu plus juste des capacités et des connaissances d'un individu à partir d'un nombre réduit de questions.

On pourra donc mieux guider son apprentissage, car dès lors qu'une connaissance semble acquise ou, au contraire, dès que l'on détecte clairement une lacune, on pourra ensuite ignorer cette compétence ou cette connaissance et évaluer le reste des points à évaluer.

D'autre part, d'un point de vue pratique, un même élève pourra repasser le questionnaire autant de fois qu'il le voudra dès lors qu'on prend soin de modifier la ou les première(s) question(s) posée(s). Dans la mesure où la banque de questions est suffisamment fournie, la probabilité qu'une même question revienne est relativement faible.

Ainsi, on disposera d'une très bonne évaluation d'un élève qui permettra de proposer par la suite un dispositif de suivi plus efficace, car basé sur une meilleure connaissance de l'individu à suivre.

3.4 Travail à effectuer

On a vu jusqu'ici l'intérêt que l'on peut avoir à mettre en place un questionnaire adaptatif. De plus, on a vu que dans la littérature, il existe de nombreuses techniques de modélisation d'un apprenant. Notre travail sera donc concentré sur la recherche de la méthode de classification la mieux adaptée aux particularités des données que nous possédons, à savoir un nombre important de questions (900 questions environ regroupées par ensembles de 60 questions) et pour chaque ensemble de questions un nombre important de répondants (300 en moyenne par questionnaire). D'autre part, il faudra trouver une méthode efficace pour le choix de l'ordre des questions.

Notre travail d'évaluation se découpe donc en trois parties : la sélection des classificateurs, le test de ces classificateurs avec différents jeux de données et l'identification d'une méthode de sélection de l'ordre des questions.

3.5 Applications client/serveur

Une fois la comparaison des classificateurs terminée, il faut penser à la mise en place du système. Comme indiqué précédemment, le client sera probablement un greffon que l'on pourra ensuite intégrer à un système existant tel que Moodle. Il faudrait donc idéalement reporter la complexité vers un serveur qui gérerait différents questionnaires et qui pourrait gérer plusieurs tests simultanément par exemple si le test est effectué en classe.

Lors du choix du langage de développement du serveur, il ne faudra pas perdre de vue deux éléments.

D'une part, les différents classificateurs existants peuvent avoir une implémentation existante en Java, notamment pour tous les classificateurs du banc d'essai *WEKA*, ou être développés dans le langage spécialisé dans le traitement de données et d'analyse statistique R, comme *POKS* par exemple. Le serveur que l'on mettra en place doit donc pouvoir être interfacé avec Java ou avec R. Bien sûr, si des classificateurs existants sont écrits dans d'autres langages, il faudra les prendre en compte.

D'autre part, afin de ne pas mettre trop de contraintes pour la mise en place d'un tel serveur, il faut idéalement le concevoir dans un langage qui peut être exécuté avec n'importe quel système d'exploitation (notamment Microsoft Windows et GNU/Linux).

Le client quant à lui ne sera développé que pour vérifier le fonctionnement du serveur. On ne prendra pas soin d'intégrer le client dans un environnement existant ni de fournir toutes les fonctionnalités qui ne dépendent pas du serveur, par exemple le choix des thèmes ou les différentes conditions d'arrêt du questionnaire, tels qu'un temps limité, un nombre maximum de questions, etc. Il ne s'agira donc en tout et pour tout que d'une preuve de concept qui nécessiterait plus de travail pour une mise

en production.

Chapitre 4

Approches d'apprentissage machine

La première étape est le choix des classificateurs. Pour cela, nous avons pris en compte les conclusions des articles indiqués précédemment afin d'avoir une présélection qui fournit des classificateurs qui ont été jugés performants en fouille de données et qui couvrent des approches différentes et complémentaires. Les classificateurs que nous avons sélectionnés sont donc les suivants :

- *zero-rule*, un classificateur sans inférence qui nous servira de référence ;
- un arbre de décision construit à l'aide d'un algorithme proche de C4.5 ;
- le classificateur bayésien naïf ;
- un réseau bayésien ;
- *POKS*, définit dans Desmarais *et al.* (2006a) et qui semble donner de bons résultats pour l'évaluation d'apprenants.

Nous les présentons plus en détail dans les paragraphes suivants. Notamment, nous donnons un exemple de construction pour la question 1 présentée dans le tableau 4.1 (une réponse correcte est indiquée par V alors qu'une réponse erronée est indiquée par F).

4.1 Classificateur sans inférence

Afin de pouvoir effectuer des comparaisons plus efficaces qu'avec le simple hasard, on introduit un classificateur sans inférence qu'on nommera *zero-rule* et dont le fonc-

TABLEAU 4.1 Exemple de données à classifier avec 7 répondants

Question	Rép1	Rép2	Rép3	Rép4	Rép5	Rép6	Rép7
Q1	V	F	V	V	V	F	V
Q2	F	F	V	V	V	V	V
Q3	V	F	V	F	V	F	V
Q4	F	F	V	F	V	F	V
Q5	V	V	F	F	V	V	V

tionnement est le suivant : lorsque l'on veut prédire le résultat d'une question Q_i, on choisit la classe majoritaire, c'est-à-dire que si la majorité des répondants ont répondu juste à la question Q_i, on prédit une réponse juste et au contraire si la majorité des élèves s'est trompé, on prédit une réponse erronée.

Le taux de classification correcte pour le classificateur, c'est-à-dire le nombre de cas bien classifiés sur le nombre total de cas, de chaque question est indépendant du nombre de questions connues. Autrement dit, si une question a en moyenne 7 réponses correctes sur 10, la question correspondante sera bien prédite 70% des fois en moyenne.

Si on prend l'exemple présenté dans le tableau 4.1, le classificateur de la question Q1 serait : `Vrai`, car on compte une majorité de cinq bonnes réponses pour deux mauvaises.

4.2 Arbre de décision

Le classificateur suivant sélectionné est l'arbre de décision car celui-ci présente un certain nombre d'avantages indiqués dans les articles présentés au chapitre précédent. Entre autres, l'arbre de décision fournit une représentation graphique du classificateur qui permet à la fois d'en simplifier la présentation mais aussi la compréhension. De plus, l'arbre de décision est le résultat d'une sélection de caractéristiques qui assure

l'efficacité du classificateur avec un grand nombre de caractéristiques. Dans notre cas, chaque question correspond à une caractéristique et on a donc un classificateur théoriquement performant avec un grand nombre de questions.

La technique de construction que nous utilisons est celle du banc d'essai WEKA (voir Witten et Frank (2005)) et est nommée par ses auteurs *J4.8*. Elle est très similaire à l'algorithme plus connu *C4.5*.

La construction de l'arbre est effectuée de façon récursive. En effet, l'opération du choix de la racine est ensuite répétée à chaque étape. Pour choisir la racine, on teste l'ensemble des choix possibles et on calcule le gain d'entropie apporté par chaque choix de racine possible. La décision qui composera la racine est alors celle qui a le gain d'entropie le plus grand.

Le gain d'entropie évoqué ici est la différence entre l'entropie avant le choix de la racine et l'entropie après le choix testé. L'entropie (de Shannon) correspond intuitivement à l'information apportée par le classificateur. Cela permet de mesurer l'intérêt que l'on a à effectuer ou non un choix de racine particulier. Si l'entropie est grande, cela signifie que la connaissance est faible. Cette entropie se calcule à partir des probabilités de chaque résultat possible. Dans notre cas où les seules décisions possibles sont `Vrai` et `Faux`, l'entropie totale est :

$$E = -P[X = V] \times \log_2(P[X = V]) - P[X = F] \times \log_2(P[X = F])$$

Prenons l'exemple présenté dans le tableau 4.1. L'information associée à la question 1 (5 réponses correctes et 2 incorrectes) est : $I(5,2) = E(5/7, 2/7) = -5/7 \times \log_2(5/7) - 2/7 \times \log_2(2/7) = 0,86$.

Le tableau 4.2 donne les décisions possibles ainsi que l'information correspondante et le gain d'entropie.

TABLEAU 4.2 Exemple de calcul pour la construction d'un arbre de décision

Réponse	Résultats	Entropie partielle	Entropie totale	Gain d'entropie
Q2=V	4V et 1F	0,72		
Q2=F	1V et 1F	1	0,8	0,06
Q3=V	4V	0		
Q3=F	1V et 2F	0,92	0,39	0,47
Q4=V	3V	0		
Q4=F	2V et 2F	1	0,57	0,29
Q5=V	3V et 2F	0,97		
Q5=F	2V	0	0,69	0,17

Au final, la question qui apporte le plus d'information est la question 3, c'est donc celle-ci qui sera la racine de l'arbre. On répète cette opération récursivement pour obtenir l'arbre de la figure 4.1.

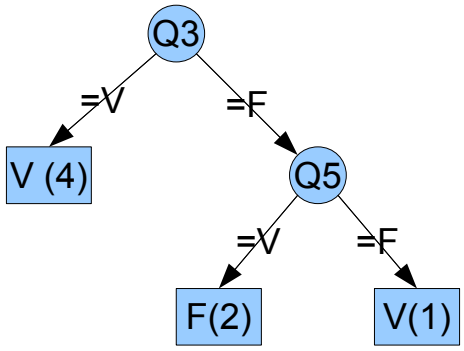


FIGURE 4.1 Arbre de décision prédisant la réponse à la question 1

Les nombres indiqués entre parenthèse indiquent les effectifs de chaque feuille. Par exemple, on voit que si la réponse à la question 3 est correcte, on peut prévoir une réponse correcte à la question 1. Les statistiques révèlent quatre cas correspondants à cette situation. Dans le cas où la réponse à la question 3 est erronée, mais que celle de la question 5 est correcte, on prévoit une réponse incorrecte à la question 1, mais

dans ce cas, la prédiction n'est appuyée que par 2 occurrences dans les statistiques.

Avec des données plus grandes, on imposera une taille minimale à chaque effectif afin de ne pas sur-entraîner le modèle. On limitera la profondeur de l'arbre pour les mêmes raisons.

Dans le cas où la réponse à certaines questions n'est pas encore connue, on considère que les deux cas sont possibles et on fait la somme des effectifs des deux cas possibles. Si avec l'exemple précédent on ne connaît pas la réponse à la question Q3 et que la question Q5 a été réussie, on trouvera alors que la question a été réussie 4 fois et échouée 2 fois. Les probabilités sont finalement $Pr[Q_1 = V] = \frac{4}{2+4} = \frac{4}{6} = 0,67$ et $Pr[Q_1 = F] = \frac{2}{2+4} = \frac{2}{6} = 0,33$. On prédira donc que la question sera réussie.

4.3 Classificateur bayésien naïf

Les classificateurs bayésiens sont très répandus dans le domaine de l'apprentissage machine. Nous en présentons deux parmi les mieux connus : les classificateurs bayésiens naïfs et les réseaux bayésiens. Commençons par celui qui en est à la base, à savoir le classificateur bayésien naïf. Nous présentons les réseaux bayésiens dans la section qui suit.

L'idée derrière un classificateur bayésien naïf est de considérer que chaque réponse a une influence égale sur la classe que l'on cherche à prédire et que les attributs sont indépendants entre eux. Ce n'est pas exact car les questions ne sont pas indépendantes mais cela permet en pratique d'avoir un schéma simple qui fonctionne bien dans la plupart des cas (voir Domingos et Pazzani (1997)).

Cette méthode est basée sur la règle de Bayes de probabilité conditionnelle. En considérant l'hypothèse H et les observations E, on a :

$$Pr[H|E] = \frac{Pr[E|H] \cdot Pr[H]}{Pr[E]}$$

Pour nos besoins, l'hypothèse H sera le succès ou l'échec à un item dont on a pas encore observé la réponse, et l'évidence E représentera les succès ou échecs observés pour d'autres items.

TABLEAU 4.3 Nouveau répondant pour le questionnaire-exemple

Q1	Q2	Q3	Q4	Q5
?	V	F	F	V

Considérons un nouveau répondant tel que présenté dans le tableau 4.3 et en reprenant les données du tableau 4.1. Prenons que l'hypothèse testée H est $Q_1 = V$. L'évidence E sera une combinaison de toutes les autres réponses données par ce nouvel arrivant, à savoir $Q_2 = V$, $Q_3 = F$, $Q_4 = F$ et $Q_5 = V$. Comme les 4 évidences, que l'on notera E_2, E_3, E_4 et E_5, sont indépendantes, la probabilité de la combinaison est le produit des probabilités. On a alors :

$$Pr[H|E] = \frac{Pr[E_2|H] \cdot Pr[E_3|H] \cdot Pr[E_4|H] \cdot Pr[E_5|H] \cdot Pr[H]}{Pr[E]}$$

Les probabilités sont issues des données connues. En effectuant les calculs et les

mêmes opérations pour l'hypothèse H' $Q_1 = F$, on obtient :

$$\begin{aligned}
Pr[Q_1 = V] &= 5/7 \\
Pr[Q_2 = V|Q_1 = V] &= 4/5 \\
Pr[Q_3 = F|Q_1 = V] &= 1/5 \\
Pr[Q_4 = F|Q_1 = V] &= 2/5 \\
Pr[Q_5 = V|Q_1 = V] &= 3/5 \\
Pr[Q_1 = V|E] &= \frac{4/5 \cdot 1/5 \cdot 2/5 \cdot 3/5 \cdot 5/7}{Pr[E]} = \frac{0,02743}{Pr[E]}
\end{aligned}$$

$$\begin{aligned}
Pr[Q_1 = F] &= 2/7 \\
Pr[Q_2 = V|Q_1 = F] &= 1/2 \\
Pr[Q_3 = F|Q_1 = F] &= 1 \\
Pr[Q_4 = F|Q_1 = F] &= 1 \\
Pr[Q_5 = V|Q_1 = F] &= 1 \\
Pr[Q_1 = F|E] &= \frac{1/2 \cdot 1 \cdot 1 \cdot 1 \cdot 2/7}{Pr[E]} = \frac{0,14286}{Pr[E]}
\end{aligned}$$

Comme de plus $Pr[H|E] + Pr[H'|E] = 1$, on a donc :

$$P[E] = Pr[H|E] \cdot Pr[E] + Pr[H'|E] \cdot Pr[E]$$

et au final :

$$\begin{aligned} Pr[E] &= 0,02743 + 0,14286 = 0,17029 \\ Pr[Q_1 = V|E] &= \frac{0,02743}{0,17029} = 16,1\% \\ Pr[Q_1 = F|E] &= \frac{0,14286}{0,17029} = 83,9\% \end{aligned}$$

La probabilité que la réponse soit fausse étant supérieure à la probabilité que la réponse soit juste, on va donc prévoir une mauvaise réponse à la question Q_1.

Cet exemple peut facilement se généraliser à un plus grand nombre de questions, il suffit de multiplier par les probabilités conditionnelles correspondantes. De même, la gestion des données manquantes est aisée car il suffit de supprimer le terme correspondant de la multiplication. Si dans l'exemple précédent la question Q_5 n'avait pas encore été posée, on aurait :

$$\begin{aligned} Pr[Q_1 = V|E] &= \frac{4/5 \cdot 1/5 \cdot 2/5 \cdot 5/7}{Pr[E]} = \frac{0,04571}{Pr[E]} \\ Pr[Q_1 = F|E] &= \frac{1/2 \cdot 1 \cdot 1 \cdot 2/7}{Pr[E]} = \frac{0,14286}{Pr[E]} \\ Pr[Q_1 = V|E] &= \frac{0,04571}{0,04571 + 0,14286} = 24,2\% \\ Pr[Q_1 = F|E] &= \frac{0,14286}{0,04571 + 0,14286} = 75,8\% \end{aligned}$$

Au final, on a un classificateur facile à comprendre et à mettre en $\frac{1}{2}$uvre et qui permet bien souvent d'obtenir de meilleurs résultats que des classificateurs plus complexes.

Remarque sur l'estimation de Laplace

On peut remarquer que pour un nombre faible de données, certaines probabilités risquent d'être nulles, sans pour autant que cela soit significatif et par conséquence la multiplication produira un résultat égal à zéro sans que cela ne soit réellement justifié.

Pour éviter cela, au lieu d'estimer la probabilité d'une décision simplement avec les effectifs, on va ajouter un terme constant qui permet d'avoir des probabilités éventuellement faibles mais en tout cas non nulles.

Par exemple, si pour une question Q_i, le nombre de répondants est de n, que n_V personnes ont correctement répondu et que n_F personnes ont échoué, les probabilités estimées ne seront pas :

$$P[Q_i = V] = \frac{n_V}{n_V + n_F}$$
$$P[Q_i = F] = \frac{n_F}{n_V + n_F}$$

mais :

$$P[Q_i = V] = \frac{n_V + \mu/2}{n_V + n_F + \mu}$$
$$P[Q_i = F] = \frac{n_F + \mu/2}{n_V + n_F + \mu}$$

On prendra arbitrairement $\mu = 3$ dans notre cas et on utilisera cette estimation de la probabilité dans le reste de l'étude.

4.4 Réseau bayésien

Le classificateur naïf fournit une façon efficace de calculer les probabilités de chaque décision mais ne permet pas de prendre en compte des relations pouvant exister entre les attributs et ne permet pas de représenter les données graphiquement comme le permet l'arbre de décision, ce qui permettrait de comprendre plus aisément les relations qui peuvent exister entre les questions. Le réseau bayésien permet de reprendre les qualités de chacun en représentant le classificateur par un graphe dirigé.

Le réseau bayésien se décompose en deux parties distinctes : le graphe orienté dont les n$\frac{1}{2}$uds correspondent aux variables à prédire (pour nous les réponses aux questions) et les arcs correspondent aux relations de dépendances, et les distributions locales de probabilité définies pour chaque n$\frac{1}{2}$ud et qui se présente sous la forme d'une table de probabilité P[n$\frac{1}{2}$ud/parents(n$\frac{1}{2}$ud)].

La construction d'un réseau bayésien n'est pas aussi simple que celle des autres classificateurs présentés précédemment. Typiquement, cette phase de construction peut soit être laissée à la charge d'un expert ou être effectuée automatiquement par recherche d'un optimal dans un espace de recherche possible souvent très large. Il faut donc définir deux éléments : la fonction d'évaluation d'un graphe et la technique de recherche. Pour notre étude, on choisit de prendre un algorithme de recherche de type K2 (voir Cooper et Dietterich (1992)) et une fonction d'évaluation simple : la vraisemblance.

La fonction d'évaluation comprend en général une composante liée à la force des liens et une autre basée sur la probabilité globale de la topologie. Pour plus de détails sur l'implémentation d'un tel réseau, on se référera à Witten et Frank (2005).

La phase d'utilisation du réseau utilise les mêmes principes d'inférence que ceux

utilisés pour un bayésien naïf sauf que seules les probabilités issues des n$\frac{1}{2}$uds parents et du n$\frac{1}{2}$ud cible sont utilisés. On utilisera tout simplement les distributions définies précédemment pour effectuer cette inférence.

Durant la phase de calcul des probabilités conditionnelles on utilisera à nouveau l'estimé de Laplace évoqué dans la section 4.3 afin de ne pas avoir d'erreurs liée au nombre faible de données.

Le résultat pour notre exemple est présenté dans la figure 4.2. Intuitivement, le réseau obtenu nous montre que les probabilités de bonne réponse aux questions 3 et 4 sont influencées par les réponses aux questions 1 et 5 et par les réponses aux questions 3 et 4 respectivement alors que les probabilités de bonne réponse aux questions 1, 2 et 5 dont indépendantes des autres réponses.

Le calcul des probabilités $Pr[Q_1 = V|E]$ et $Pr[Q_1 = F|E]$, où E représente toujours les évidences présentées dans la table 4.3, s'effectue alors comme suit :

$$\begin{aligned}
Pr[Q_1 = V|E] &= \frac{0,125 \cdot 0,69}{Pr[E]} = \frac{0,04571}{Pr[E]} \\
Pr[Q_1 = F|E] &= \frac{0,833 \cdot 0,31}{Pr[E]} = \frac{0,14286}{Pr[E]} \\
Pr[Q_1 = V|E] &= \frac{0,08625}{0,08625 + 0,25823} = 25,0\% \\
Pr[Q_1 = F|E] &= \frac{0,25823}{0,08625 + 0,25823} = 75,0\%
\end{aligned}$$

On a donc les probabilités de chaque décision et on peut conclure que la réponse à la question Q_1 sera erronée.

Comme on peut le voir, les réseaux bayésiens sont des outils puissants car ils permettent de calculer des probabilités pour la prédiction et permettent d'avoir une représentation graphique claire du classificateur mais ils n'ont par contre pas de méthode facile de construction et ont donc un temps de construction plus long que les

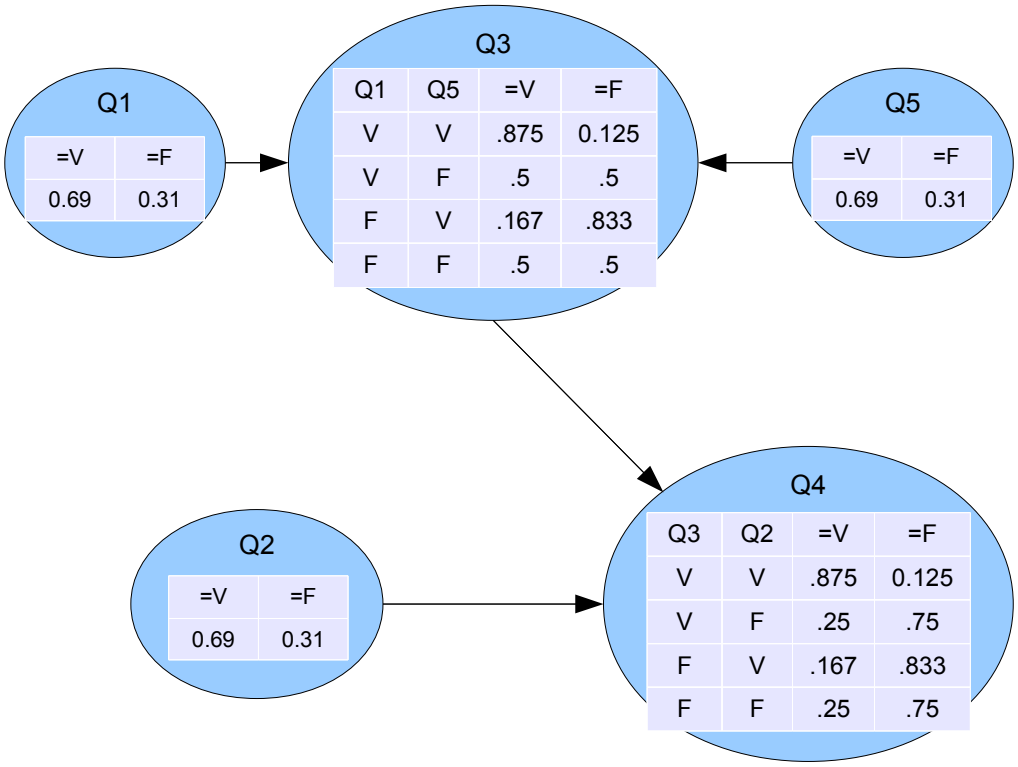


FIGURE 4.2 Réseau bayésien du problème à 5 questions

autres classificateurs considérés jusqu'ici.

4.5 Réseau *POKS*

Pour finir la revue des classificateurs que l'on a choisis, nous allons expliciter le classificateur *POKS* présenté dans Desmarais *et al.* (2006a). *POKS* signifie *partial order knowledge structures*, autrement dit, structure de connaissances avec ordre partiel.

Le réseau *POKS* part de l'idée qu'il existe une hiérarchie entre les différentes questions telle que si certaines questions sont réussies, cela implique que les questions plus « simples » seront également réussies et que d'un autre côté, si une question est échouée, toutes les questions plus « difficiles », c'est-à-dire dont la compétence nécessaire pour répondre correctement est supérieure, seront également échouées. Afin de représenter un questionnaire, on utilisera comme pour le réseau bayésien des graphes orientés acycliques dont les flèches symboliseront la relation de dépendance « plus difficile ». Contrairement au réseau bayésien pour lesquels les arcs du graphe ne symbolisent qu'une relation de causalité (ou n'importe quelle relation probabiliste), les arcs du graphe *POKS* représentent un pré-requis et cela implique que dans une relation $A \succ B$, la probabilité $P(B|A) \approx 1$ et que la question A soit réussie est nécessairement inférieure ou égale à la probabilité que la question B soit réussie.

La construction d'un tel réseau peut être effectué de plusieurs manières différentes et il s'agit d'un domaine de recherche toujours actif. La première technique développée dans Kambouri *et al.* (1994) est semi-automatique dans la mesure où l'algorithme interroge un expert afin de déterminer les relations entre certaines questions. Une autre technique, développée dans Desmarais et Gagnon (2006) cherche à déterminer

si la relation $A \succ B$ est vraie en testant les trois conditions suivantes :

$$\begin{aligned} Pr[B|A] &\geq p_c \\ Pr[\overline{A}|\overline{B}] &\geq p_c \\ Pr[B|A] &\neq Pr[B] \end{aligned}$$

La première ligne teste si B est réussie sachant que A est réussie. La seconde teste si un échec de A entraîne un échec de B. Et enfin la dernière condition vérifie que A a bel et bien une influence sur B. Ces trois conditions sont vérifiés avec un test statistique en prenant $p_c = 0,5$ pour les deux premières conditions et en effectuant un test d'indépendance pour la troisième condition.

L'inférence s'effectue de la façon suivante en considérant que H est l'hypothèse testée et $E_1, \ldots, E_n$ sont les évidences qui possèdent une relation $E_i \succ H$:

$$\begin{aligned} O[H|E_1, \ldots E_n] &= O[H] \prod_i \frac{Pr[E_i|H]}{Pr[E_i|\overline{H}]} \\ O[X] &= \frac{Pr[X]}{1 - Pr[X]} \\ Pr[X] &= \frac{O[X]}{1 + O[X]} \end{aligned}$$

L'exemple que nous avons pris jusqu'ici pour montrer le fonctionnement des classificateurs ne fonctionne pas réellement pour le classificateur *POKS*, car le nombre de questions est faible et le réseau créé ne contiendrait qu'un seul lien (de Q_3 vers Q_4). Pour mieux comprendre le principe, on peut considérer l'exemple de la figure 4.3. Les connaissances (b) et (c) nécessite la maîtrise de la connaissance (d). La maîtrise de la connaissance (a) nécessite la maîtrise des connaissances (b) et (c). Pour des détails plus précis sur l'implémentation, on se reportera à Desmarais et Gagnon (2006).

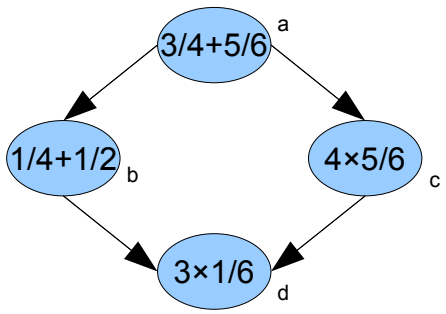


FIGURE 4.3 Réseau *POKS* pour le calcul fractionnaire

Pour conclure sur le choix de *POKS*, il semble présenter de meilleurs résultats que les réseaux bayésiens dans le cas où les données représentées sont des questions d'un questionnaire et on peut donc espérer avoir de bons résultats pour le projet qui nous intéresse (voir Desmarais *et al.* (2006b)).

Chapitre 5

Expériences et méthodologie

Afin d'avoir le système d'évaluation le plus efficace possible, il faut sélectionner le classificateur le plus capable de prédire les réponses aux questions futures. Pour cela, on divise le travail expérimental tel que suit :

- le test des classificateurs avec une seule question manquante ;
- la détermination de l'ordre optimal ;
- la comparaison des classificateurs avec un comportement proche de la réalité (nombre de questions manquantes décroissant avec ordre aléatoire puis optimal) ;
- des tests supplémentaires si nécessaires, notamment en découpant les questions par thème.

On s'attardera par la suite sur l'explication des différents résultats obtenus lors des tests et les performances des classificateurs seront jugés sur cette base.

5.1 Présentation des données

Les classificateurs ont été testés sur quatre jeux de données différents issus de questionnaires réellement soumis à des humains :

1. **Test de commandes shell UNIX (`unix`)**. Ce sont les résultats d'un test posé à 48 répondants et contenant 34 questions portant sur différentes commandes de terminal Unix, allant de commandes simples (ex : `cd`, `ls`) à des plus complexes

(ex : `awk`, `sed`), en passant par des commandes d'administration système (ex : `ps`, `chgrp`).

2. **Test de français (`français`)**. Ce jeu de données contient 41 répondants à 160 questions d'un test de français posé à des immigrants.

3. **Test d'arithmétique (`vomlel`)**. Vomlel Vomlel (2004) a réuni les réponses de 149 jeunes élèves à 20 questions d'un test portant sur des calculs simples de fractions.

4. **Test de mathématiques de Polytechnique (`maths`)**. Issus des résultats des élèves de première année de baccalauréat de l'École Polytechnique de Montréal de juin 2000, ces données se répartissent en 453 élèves et 60 questions. On se réfèrera à l'annexe C pour avoir des exemples de questions.

Le tableau 5.1 résume ce qui est précisé précédemment et donne le taux de réussite moyen de l'ensemble des élèves pour chaque questionnaire.

TABLEAU 5.1 Résumé des jeux de données

Jeu de données	Nb. de questions	Nb. de répondants	Taux de réussite moyen
`unix`	34	48	53%
`français`	160	41	57%
`vomlel`	20	149	61%
`maths`	60	453	57%

Pour chaque jeu de données, les expérimentations ont été faites suivant les modalités suivantes : on retire un élève du jeu de données, on entraîne le modèle avec le reste des données et on teste sur l'élève que l'on a retiré. On répète ce procédé pour chaque élève du jeu de données.

Nous allons à présent présenter les différentes expériences que nous avons menées en expliquant l'objectif poursuivi par chacune des expériences.

5.2 Tests préliminaires

Avant tout, il nous semble opportun de comparer les différents classificateurs que nous avons sélectionnés dans un cas simple, c'est-à-dire sans questions inconnues parce que encore non posées. Autrement dit, on cherche à comparer la performance avec les réponses à toutes les questions connues sauf la réponse à une question que l'on cherche à prédire.

Afin de connaître la performance moyenne sur l'ensemble des questions, nous avons répété l'expérience pour chaque classe. De même, afin d'avoir un résultat pertinent, cette expérience a été répétée en séparant les données de la manière suivante : d'un côté un ensemble d'entraînement qui est composé de 90% des répondants et de l'autre des données de test qui sont composées des 10% restants. On répète cela 10 fois afin d'avoir une performance moyenne. Cette technique classique est appelée une validation croisée à 10 replis.

L'algorithme de test se décrit de la façon suivante : Soit Q l'ensemble des questions et R l'ensemble des répondants. On partitionne R en 10 sous-ensembles R_i, $i \in \{1 \dots 10\}$. Pour chaque sous-ensemble R_i, on effectue les opérations suivantes. Le sous-ensemble R_i est l'ensemble de test et l'ensemble $R - \{R_i\}$ est l'ensemble d'entraînement des classificateurs. Pour chaque Qi de Q, on suppose que la réponse est connue pour toutes les questions restantes $Q - \{Qi\}$ et on entraîne le classificateur en utilisant les données de $R - \{R_i\}$. On calcule, pour la question Q_i, le nombre de prédictions correcte dans l'ensemble R_i. On répète cela pour chaque question et chaque sous-ensemble de l'ensemble des répondants.

5.3 Comparaison avec des questions non répondues

Une fois ce premier test effectué, il parait opportun de tester le pouvoir de prédiction avec un certain nombre de questions non répondues. Pour cela, on reprend le principe du test précédent sauf que plutôt que de faire une validation croisée avec 10 groupes, on entraîne les classificateurs avec l'ensemble des répondants sauf un, pour ensuite tester le classificateur sur ce dernier. Cette technique est connue en anglais sous le nom de *leave-one-out validation* et permet de réutiliser le plus de données d'entraînement possible. De plus, on effectuera les expérimentations avec un ordre de réponse aléatoire afin d'avoir une moyenne plutôt que de risquer d'avoir un ordre de questions particulièrement efficace ou au contraire particulièrement mauvais.

L'algorithme de simulation du processus de questions-réponses devient alors, pour chaque répondant : Soit Q l'ensemble des questions et R l'ensemble des répondants. Pour chaque répondant R_i, on effectue les opérations suivantes : Le sous-ensemble $\{R_i\}$ est l'ensemble de test et l'ensemble $R - \{R_i\}$ est l'ensemble d'entraînement des classificateurs. On considère l'ensemble des questions connues Q_c qui est initialement vide. On sélectionne aléatoirement une question de $Q - Q_c$ pour laquelle on entraîne les classificateurs avec l'ensemble d'entraînement $R - \{R_i\}$. On vérifie alors si la prédiction pour R_i avec seulement les questions connues est correcte ou non puis on réitère le processus à partir du choix aléatoire de la question. On prendra naturellement soin d'ajouter les questions sélectionnées à Q_c au fur et à mesure. On répète cela jusqu'à ce que $Q = Q_c$ et pour l'ensemble des répondants.

Les résultats que l'on obtient alors correspondent aux résultats que l'on obtiendrait si on essayait les classificateurs sur un questionnaire non adaptatif pour lequel l'ordre des questions n'est pas adapté aux réponses de l'élève. Mais ce n'est pas encore ce que l'on veut puisqu'on veut pouvoir choisir les questions de façon plus « intelligente ». On

va donc s'attarder dans l'expérience suivante à vérifier quelle est la meilleure méthode pour déterminer la question suivante.

5.4 Détermination de l'ordre des questions

Les expériences précédentes adoptent un ordre de choix de question systématique ou aléatoire. Or, la force des tests adaptatifs est justement de déterminer un ordre optimal pour en arriver à une évaluation de la connaissance de l'apprenant. Ce ne sont pas tous les modèles qui permettent de déterminer cet ordre et il est plausible que leur performance diffère selon le choix de l'ordre. Une autre expérience consiste donc à déterminer quel ordre de questions paraît la plus pertinente. Pour cela, on proposera un ordre possible et on vérifiera si l'amélioration apportée par ce nouvel ordre le justifie.

Il s'agit donc du même algorithme que précédemment mais en remplaçant « le choix de Q_i » par « le choix de Q_i qui maximise la performance ».

5.5 Paramètres des classificateurs

Dans cette section nous présentons les paramètres utilisés pour chaque classificateur.

Classificateur sans inférence

Ce classificateur est très simpliste, il ne nécessite donc pas de paramètres particuliers.

Arbre de décision

Comme précisé dans le chapitre précédent, on utilise un algorithme de construction *J48*, implémentation du banc d'essai WEKA de l'algorithme *C4.5*. Chaque feuille doit contenir au minimum un effectif de 2 cas afin d'éviter le sur-échantillonage.

Classificateur bayésien naïf

Ce classificateur ne nécessite aucun ajustement particulier. À noter simplement l'utilisation de l'estimé de Laplace décrit précédemment.

Réseaux bayésiens

Pour la construction du réseau bayésien, on utilise un algorithme de recherche local K2 avec un état initial vide, c'est à dire un réseau sans liens. On spécifie de plus que le nombre maximal de parents pour chaque n$\frac{1}{2}$ud doit être de 3.

Classificateur *POKS*

Les paramètres utilisés pour la construction du réseau *POKS* sont les suivants (voir Desmarais *et al.* (1995) pour l'explication de chaque paramètre) :

$$\alpha_c = 0,15$$

$$\alpha_p = 0,2$$

Chapitre 6

Résultats et discussion

Dans ce chapitre, nous présentons les différents résultats obtenus. Nous développons les résultats et nous expliquons leur provenance. Nous commençons par l'expérience statique avant de continuer par le choix des questions, l'expérience dynamique et de finir par les expériences complémentaires que nous avons pu effectuer.

6.1 Comparaison préliminaire des classificateurs

La première expérience nous permet de faire une comparaison rapide des différents classificateurs choisis.

Les résultats que l'on obtient lors de cette expérience sont compilés dans le tableau 6.1. Ces résultats correspondent au nombre de cas pour lesquels la classification donnée par les classificateurs (réponse correcte ou erronée) est la même que le résultat réel observé par rapport au nombre d'essais total.

Tableau 6.1 Résultats de la comparaison statique

Questionnaire	Zero-rule	J48	Bayésien naïf	Réseau bayésien	POKS
`unix`	72,37%	84,25%	83,88%	83,76%	83,74%
`vomlel`	73,96%	87,35%	81,64%	85,97%	86,42%
`français`	68,17%	67,13%	71,71%	62,41%	74,61%
`maths`	70,89%	69,06%	69,83%	70,58%	73,21%

Ce tableau nous permet de noter plusieurs choses. Tout d'abord, on vérifie que chacun des cinq classificateurs sélectionnés permet d'obtenir des résultats intéressants

sur plusieurs jeux de données différents. Nous ne discuterons pas des performances du classificateur sans inférence *zero-rule* car celui-ci est surtout choisi pour avoir une idée des résultats minimaux qu'on est en mesure d'attendre.

On se rend compte que pour des jeux de données avec un nombre réduit de questions, tels que `unix` et `vomlel`, les résultats sont satisfaisants pour l'ensemble des classificateurs considérés. Satisfaisants car réellement supérieurs aux résultats de *zero-rule* par au moins 10%.

La deuxième remarque porte sur les résultats avec le jeu de données `français`. Celui-ci a un grand nombre de questions et un nombre de données plus faible (160 questions et 41 répondants). On se rend compte que l'arbre de décision ainsi que le réseau bayésien donnent alors de mauvais résultats, inférieurs au classificateur de base. Il faudra éventuellement s'en inquiéter si on désire ensuite supprimer le découpage en ensembles de 60 questions qui existe pour les questionnaires de mathématiques. On verra lors des prochaines expériences à quel point cela est gênant.

Enfin, on peut remarquer que les résultats obtenus pour le questionnaire de mathématiques sont très proches les uns des autres et ne sont pas meilleurs que ceux obtenus avec un classificateur basique.

On a donc deux conclusions importantes à tirer de ces résultats. Tout d'abord il faut être vigilant face aux résultats obtenus pour un grand nombre de questions car il semblerait que certains classificateurs ne fonctionnent pas aussi bien avec un grand nombre de questions qu'avec un nombre réduit de questions. Ensuite, on remarque que les résultats obtenus avec le questionnaire de mathématiques ne semblent pas identiques à ceux des autres questionnaires. Il faudra vérifier si ce comportement se poursuit avec les prochaines expériences.

À noter qu'au début de l'expérience nous avions sélectionné un classificateur supplémentaire dont nous n'indiquons finalement pas les résultats : un classificateur avec

une seule règle. En effet, le comportement de ce classificateur avec des données manquantes est très mauvais et il n'est donc pas intéressant.

Nous allons maintenant comparer l'évolution des résultats en fonction du nombre de réponses connues.

6.2 Tests avec ordre des questions aléatoire

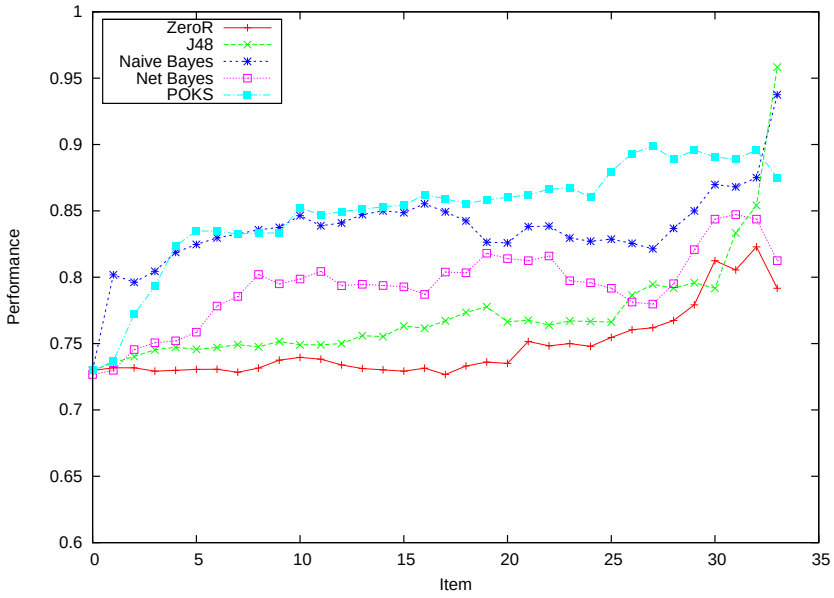
Nous avons vu dans la section précédente que les résultats obtenus lorsque l'on cherche à prédire une question quand toutes les autres sont connues sont assez équivalents avec chaque classificateur. On s'intéresse à présent au cas où un plus grand nombre de réponses est inconnu. Cette expérience, avec un ordre de questions aléatoire, permet d'avoir une idée du minimum que l'on pourra attendre pour l'optimisation de l'ordre des questions.

On effectue un test par répondant avec un ordre aléatoire différent pour chaque répondant.

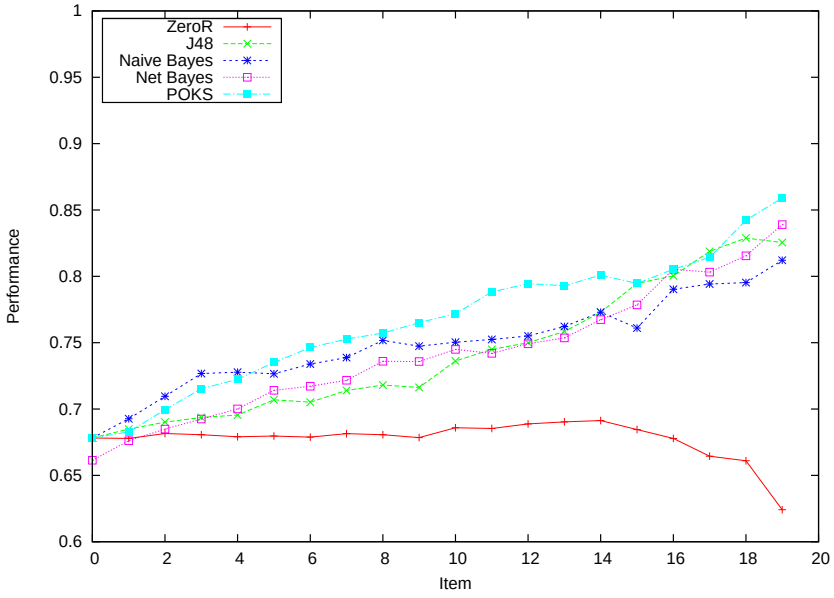
Les résultats sont consignés dans la figure 6.1. Les graphiques représentent en abscisse le nombre de questions connues et en ordonnée le pourcentage de questions restantes bien classifiées. Cette méthode de représentation des résultats cache une réalité importante : le nombre de classifications effectuées est faible lorsque le nombre de questions à classifier est faible (c'est-à-dire à la fin). Il ne faudra donc pas prendre en compte les dernières questions lors de nos observations et conclusions. Le classificateur *zero-rule* devrait ainsi avoir un pourcentage de bonne classification constant. La divergence finale montre la faible représentativité des données finales.

La performance affichée correspond au taux de question non connues dont la prédiction faite par le classificateur correspond à la réalité.

Globalement, on observe tel qu'espéré que le taux de questions correctement clas-

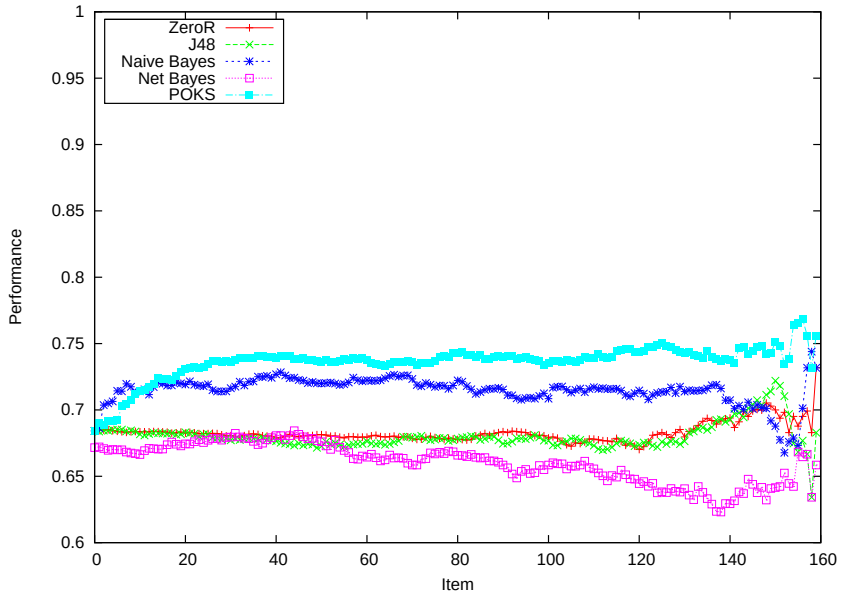


(a) Performance avec le jeu de données `unix`

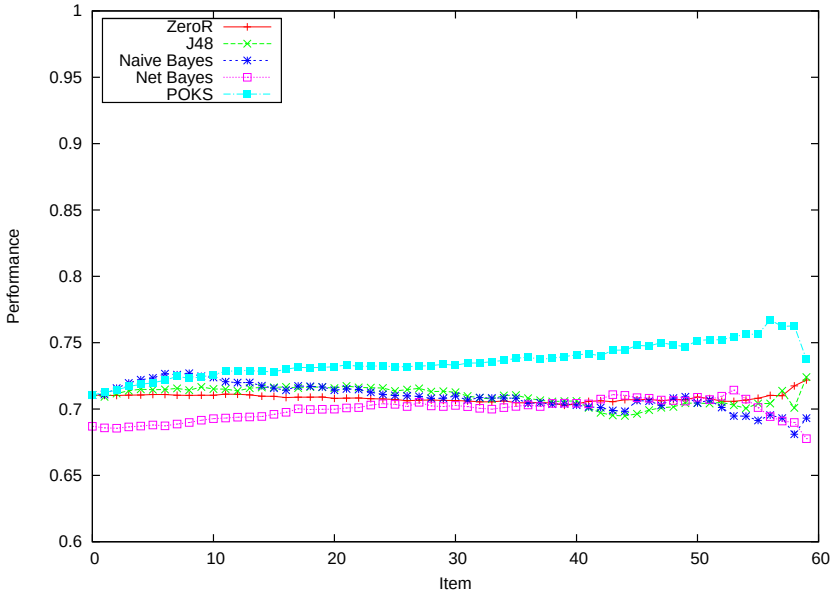


(b) Performance avec le jeu de données `vomlel`

FIGURE 6.1 Performances obtenues avec un ordre des questions aléatoire (a)



(a) Performance avec le jeu de données `français`



(b) Performance avec le jeu de données `maths`

FIGURE 6.2 Performances obtenues avec un ordre des questions aléatoire (b)

sifiées augmente avec le nombre de questions connues. Cependant, la progression n'est pas la même suivant le nombre de questions et suivant le classificateur.

Pour le questionnaire `unix`, on observe que les classificateurs *POKS* et bayésien naïf progressent rapidement avec un nombre faible de questions puis stagnent ensuite avec une performance autour de 85%. D'un autre côté, on voit que l'arbre de décision reste globalement à un niveau faible avant d'augmenter lorsque le nombre de questions connues devient important. Le réseau bayésien présente quant à lui une augmentation moins prononcée que les deux premiers classificateurs et présente un moins bon pourcentage de bonnes prédictions. On peut conclure deux points de cette observation. Tout d'abord le fait que le taux de bonnes prédictions augmente avec le nombre de questions et que les résultats soient meilleurs que le classificateur sans inférence justifie le choix d'une inférence statistique pour prévoir les résultats des élèves. D'autre part, on peut tirer une conclusion partielle sur les performances des classificateurs en indiquant que pour ce questionnaire, les classificateurs bayésien naïf et *POKS* sont plus performants que les autres. On verra ensuite s'il en est de même pour les autres questionnaires et avec un ordre optimisé.

L'autre questionnaire avec un nombre réduit de question `vomlel` ne permet pas vraiment de tirer de conclusions intéressantes. Pour tous les classificateurs avec inférence la progression est constante et de même pente. On peut tout de même noter que *POKS* a un pourcentage légèrement supérieur aux autres classificateurs mais cela n'est pas réellement significatif. On peut penser que le nombre très réduit de questions ne permet pas réellement de différencier les différents classificateurs. Il faudra vérifier si on obtient le même genre de comportement dans les tests futurs. On peut cependant observer que la performance de chaque classificateur est meilleure que celle du classificateur *zero-rule*.

Pour le questionnaire long `français`, on observe le même comportement que pour

`unix` avec les classificateurs *POKS* et bayésien naïf, à savoir une augmentation rapide pour un nombre réduit de questions puis une stagnation. Par contre le comportement des réseaux bayésiens et de l'arbre de décision nous montre que ceux-ci ne semblent pas performants avec un nombre de questions aussi important. En tout cas une hiérarchie semble se dessiner à ce niveau.

Finalement, pour le questionnaire `maths`, tous les classificateurs convergent vers la même performance, sauf *POKS* qui semble très légèrement supérieur. En tout cas la performance de l'ensemble de notre banc d'essai est proche de celle de *zero-rule*, ce qui nous indique que la classification est médiocre. Il faudra vérifier si ce comportement est toujours le même avec un ordre optimisé ou en essayant de séparer les questions en thèmes.

6.3 Détermination de l'ordre des questions

Comme on l'a vu précédemment, garder un ordre aléatoire dans les questions entraîne une stagnation dans les performances au delà d'un nombre réduit de questions (environ 1/4).

Il vient donc naturellement à l'esprit qu'en sélectionnant un ordre de questions de manière plus intelligente, on parviendra à augmenter continuellement la performance, ou en tout cas à atteindre plus rapidement de meilleurs résultats, ce qui permettra d'arrêter le questionnaire après un certain nombre de questions.

Pour déterminer cet ordre, on observe qu'il est possible de déterminer facilement l'entropie *a priori* de chaque question et de choisir alors celle qui a la plus grande entropie. En effet, si on choisit d'abord les questions avec une entropie plus grande, l'entropie générale va baisser et donc la qualité de la classification va augmenter.

L'entropie de chaque question se calcule simplement à partir de la probabilité que

la réponse soit juste $P[X = V]$. Chaque classificateur peut donner cette probabilité *a priori* et alors l'entropie est :

$$\begin{aligned} H &= -P(X = F) \cdot \log\left(P(X = F)\right) - P(X = V) \cdot \log\left(P(X = V)\right) \\ H &= -(1 - P) \cdot \log(1 - P) - P \cdot \log(P) \\ H &= -\log(1 - P) + P \cdot \log\left(\frac{1 - P}{P}\right) \end{aligned}$$

avec $P = P[X = V]$.

Il suffit alors, pour chaque choix de question, de comparer l'entropie de chaque question et de poser celle dont l'entropie est la plus grande, autrement dit celle dont la probabilité est la plus proche de 0,5, soit au final celle dont on est le moins sûr de la prédiction. La figure 6.3 montre alors le résultat que l'on obtient maintenant avec un bayésien naïf sur le questionnaire `unix` en comparaison avec les résultats obtenus pour un ordre aléatoire.

On se rend bien compte que l'ordre basé sur l'entropie permet une réelle amélioration de la prédiction et que grâce à un ordre optimisé, on peut espérer atteindre de meilleurs pourcentages de réussite plus rapidement. L'amélioration porte sur deux points : tout d'abord l'augmentation initiale est plus forte et ensuite elle dure plus longtemps. Un choix avisé des questions nous permet d'atteindre quasiment 100% de réussite pour les dernières questions.

On peut noter qu'il existe d'autres méthodes de calcul pour l'entropie, notamment avec les probabilités *a posteriori* mais la technique de calcul est lourde et ne garantit pas de meilleurs résultats (voir Desmarais et Liu (1993)).

On s'attache à présent à tester ce nouveau moyen de sélection de la prochaine question sur l'ensemble des questionnaires et avec tous les classificateurs choisis dans notre banc d'essai.

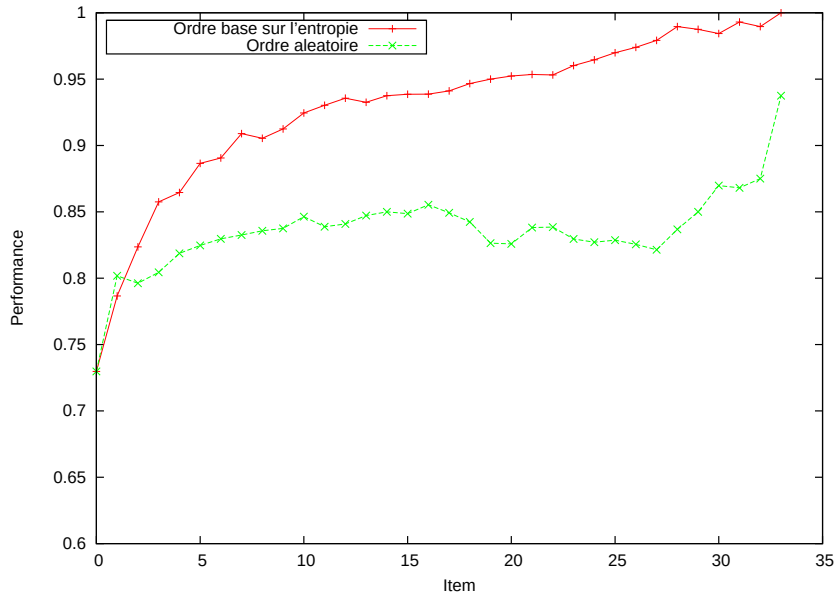


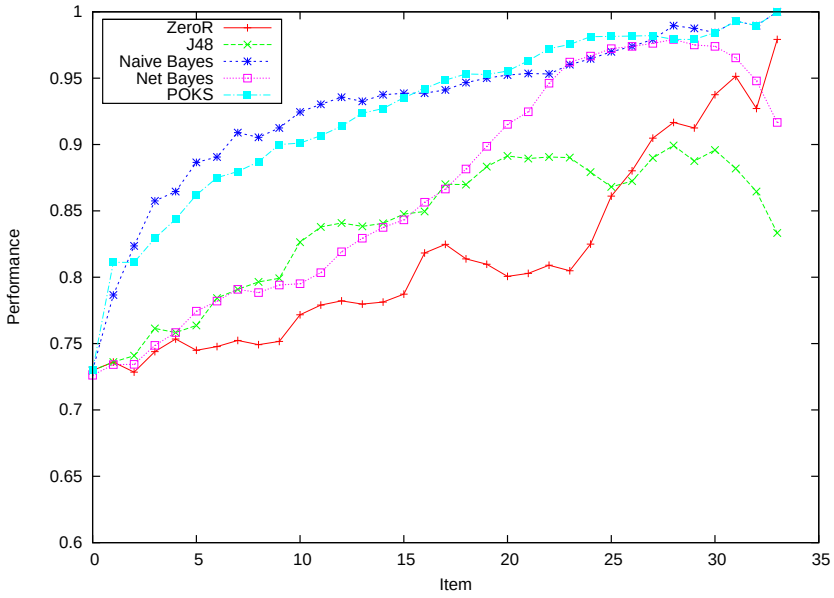
FIGURE 6.3 Comparaison entre la performance de l'ordre aléatoire et de l'ordre basé sur l'entropie avec un classificateur bayésien naïf

6.4 Comparaison en sélectionnant un ordre optimisé

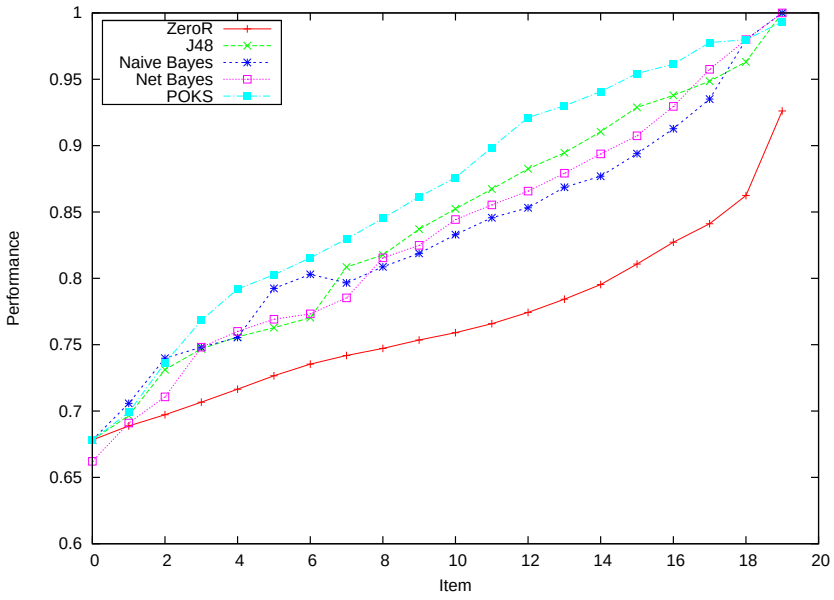
La section précédente nous convainc de l'utilité de choisir un ordre plus optimisé pour la prochaine question. Le test dont nous allons maintenant présenter les résultats permet de vérifier que cela est vrai pour tous les questionnaires et tous les classificateurs mais permet également de voir les résultats que l'on peut obtenir au final et donc de conclure définitivement sur le choix des classificateurs les plus performants pour notre questionnaire adaptatif.

On effectue un test pour chaque répondant en sélectionnant pour chaque exécution l'ordre le plus efficace en fonction des réponses connues.

La figure 6.4 présente ces résultats. On a toujours en abscisse le nombre de questions connues et en ordonnée le pourcentage de questions restantes bien classifiées. Le problème précédent de représentativité des données lorsque le nombre de questions est faible se pose à nouveau.

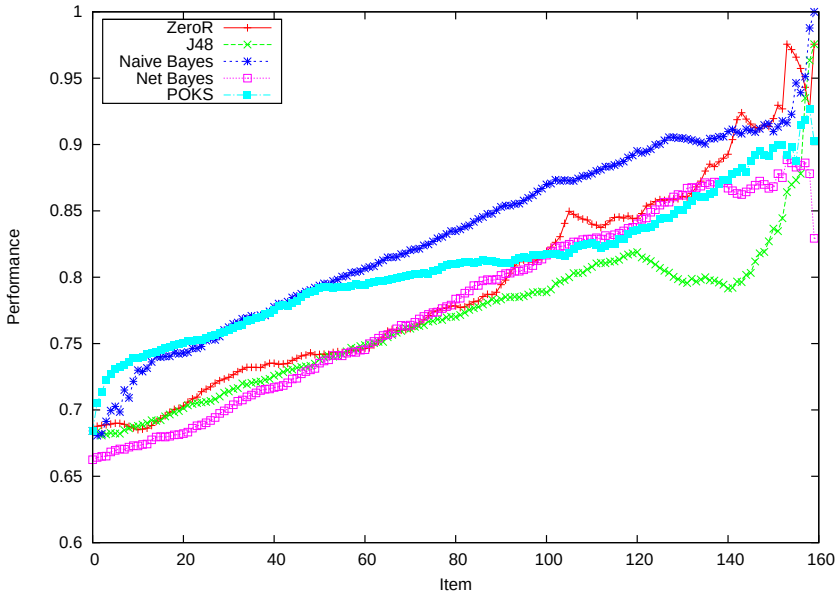


(a) Jeu de données `unix`

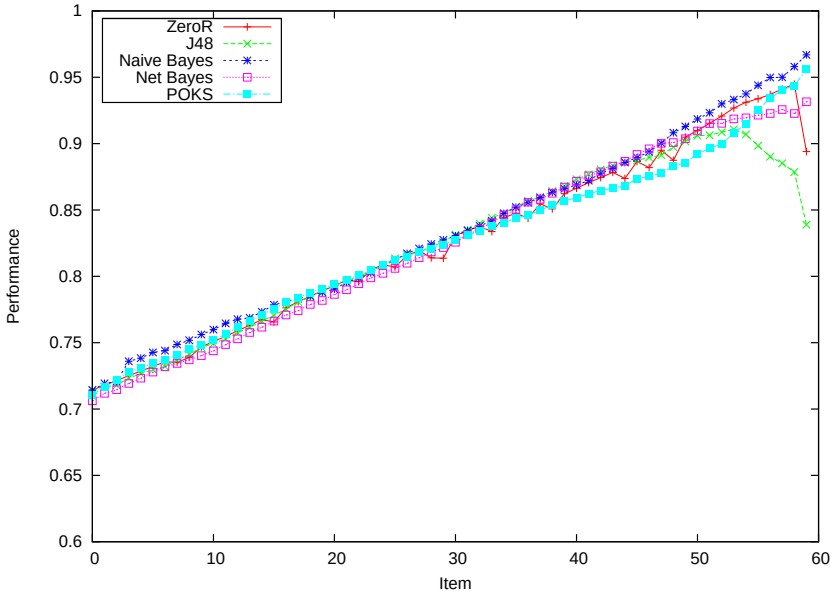


(b) Jeu de données `vomlel`

FIGURE 6.4 Performances obtenues avec un ordre basé sur l'entropie (a)



(a) Jeu de données `français`



(b) Jeu de données `maths`

FIGURE 6.5 Performances obtenues avec un ordre basé sur l'entropie (b)

Premièrement, on peux tout de suite observer que le pourcentage de classification correcte suis une pente ascendante quel que soit le classificateur et quel que soit le questionnaire. Cela justifie définitivement l'intérêt de choisir un système adaptatif.

Si on regarde maintenant individuellement chaque questionnaire, on observe un comportement différent à chaque fois, mais une hiérarchie semble tout de même se dessiner.

Si on regarde tout d'abord les résultats pour `unix`, on peut remarquer que les observations faites auparavant sont toujours valables, à savoir que *POKS* et le bayésien naïf semblent avoir un comportement très proche l'un de l'autre et que les résultats obtenus pour ces deux classificateurs sont plutôt bons car on peut alors prévoir 90% des réponses manquantes à partir de 7 questions pour le bayésien naïf et à partir de 9 questions pour *POKS*. Sur un questionnaire de 34 questions, on se trompe donc moins de 3 fois en moyenne pour 27 questions restantes après 1/5 des questions posées et connues. Pour le reste de la hiérarchie, on se rend compte que le réseau bayésien est plus performant que l'arbre de décision mais ces deux derniers ne proposent pas vraiment de bons résultats puisque qu'après la moitié des questions, on se trompe 2,5 fois en moyenne pour 17 questions restantes. Enfin, comme on s'y attendait, le classificateur sans inférence est le moins bon.

Pour le questionnaire `vomlel` la progression initiale est moins marquée mais on observe également une croissance continue du pourcentage de classification correcte. On retrouve à nouveau que le classificateur *POKS* obtient de meilleurs résultats que les autres classificateurs. Mais le nombre réduit de questions empêche de tirer des conclusions définitive sur la comparaison entre les classificateurs. Tous les classificateurs sont plus performants que le classificateur de base *zero-rule* que nous avons sélectionné pour le comparatif.

En étudiant les deux questionnaires comprenant un nombre réduit de questions

(moins de 40), on peut déjà indiquer que le classificateur *POKS* semble avoir un comportement très proche du classificateur bayésien naïf. D'autre part, l'arbre de décision ainsi que le réseau bayésien semblent légèrement mois performants.

Observons à présent le questionnaire long `français`. Sans considérer les dernières questions qui ne sont pas réellement significatives au vu du nombre réduit de répondant (41 personnes), on retrouve les considérations émises précédemment, à savoir un comportement proche pour *POKS* et le bayésien naïf et une performance médiocre pour les autres classificateurs.

Si on regarde cette fois le nombre de questions nécessaire pour atteindre 80% de bonnes classifications, on voit qu'il faut environ 50 questions connues au classificateur bayésien. En d'autres termes, au bout d'un tiers des questions, on va se tromper 20 fois pour les 110 questions restantes donc on aura 160 questions bien classifiées en divisant la durée de l'examen par 3 soit environ 7/8 des questions. Pour *POKS*, le nombre de questions nécessaires est légèrement supérieur mais reste dans le même ordre de grandeur. Cependant, pour un nombre supérieur de questions, *POKS* n'augmente pas aussi rapidement que le réseau bayésien.

Pour les autres classificateurs par contre, on se rend compte que le résultat n'est pas meilleur que *zero-rule*. Autrement dit, l'inférence n'apporte pas d'informations supplémentaires. On peut donc conclure que ces classificateurs ne seront pas efficaces dans le cadre qui nous intéresse, à savoir avec un nombre important de questions.

Arrivé à ce stade, on peux donc conclure que les classificateurs les plus efficaces dans le cas qui nous intéresse seront le classificateur bayésien naïf et le classificateur *POKS*. En effet, ils sont les plus efficaces sur un grand nombre de questions mais surtout, ils permettent une amélioration significative du taux de classification avec un nombre réduit de questions.

Finalement, afin de comparer les performances des classificateurs, nous avons aussi

mesuré la durée d'exécution du banc d'essai pour chacun des classificateurs. Ces durées d'exécution sont rassemblées dans le tableau 6.2. L'ordinateur utilisé est équipé d'un Pentium M 1,86GHz et de 1,5Gio de mémoire RAM.

TABLEAU 6.2 Durée d'exécution du banc d'essai lors des tests dynamiques

	Zero-rule	Arbre	Bayésien naïf	Réseau bayésien	POKS
`unix`	0,4 sec	1,6 sec	2,8 sec	38,0 sec	17,0 sec
`vomlel`	0,4 sec	6,2 sec	7,2 sec	51,0 sec	25,0 sec
`français`	6,2 sec	28,4 sec	137,2 sec	1215,0 sec	510,0 sec

Il faut noter que les durées présentées ici ne sont pas nécessairement représentatives car le banc d'essai effectue de nombreuses création de modèles. Or pour une application en classe, le modèle ne sera créé qu'une seule fois à partir de données d'archives et ne sera pas recréé à chaque ajout d'un élève dans les archives. Malgré cela, on se rend compte que les deux classificateurs basés sur des réseaux sont significativement plus longs à créer que les autres. De plus, on ne donne pas les temps de calcul pour le jeu de données *maths* car celui-ci est très long dans tous les cas et sa mesure n'apporte pas d'information pertinente.

Donc, sous réserve que les modèles des classificateurs soient créés a priori, le classificateur *POKS* pourra être sélectionné pour effectuer les évaluations des élèves. Dans le cas contraire, on pourra se rabattre sur le classificateur bayésien naïf dont le temps de calcul des modèles peut être compatible avec un examen en temps réel.

Enfin, pour finir sur la comparaison des classificateurs dans le cas d'un ordre déterminé dynamiquement, on remarque que les résultats obtenus avec le questionnaire `maths` ne permettent pas de conclure quoi que ce soit. En effet, tous les classificateurs semblent équivalents à un classificateur sans inférence et la situation est telle qu'on peut imaginer que toutes les questions sont indépendantes. Nous tentons dans la prochaine section de comprendre ce comportement en étudiant notamment les don-

nées utilisées et en essayant de voir si ce comportement est le même avec les données d'autres années.

6.5 Expérimentations supplémentaires

Comme on l'a vu jusqu'ici, les conclusions que nous avons pu tirer sur les différents classificateurs ne sont pas valables pour le questionnaire de mathématiques qui est pourtant celui qui nous intéresse le plus dans le cadre du projet de guide d'apprentissage à la Polytechnique. Ce questionnaire a été élaboré avec soin par des mathématiciens professionnels et comporte une banque de près de mille questions, d'où l'intérêt de pouvoir effectuer un diagnostic efficace avec un test adaptatif. Afin de comprendre ces résultats, nous avons émis plusieurs hypothèses, à la fois sur les questions mais aussi sur les réponses données par les élèves.

6.5.1 Ordre des questions du questionnaire papier

Les réponses dont on bénéficie proviennent d'un questionnaire à choix multiple proposé lors d'un test facultatif d'une durée de deux heures. Il est réaliste de penser que certains élèves abandonnent l'examen pendant la durée de celui-ci. On peut alors supposer que les questions posées à la fin de l'examen sont biaisées. Pour vérifier si tel est le cas, nous avons effectué deux études. La première consiste à calculer la moyenne obtenue pour chaque question, pour voir si une baisse est visible pour les dernières questions et la seconde consiste à comparer les réponses données à une même question pour différents questionnaires, sachant que dans les différents questionnaires, la même question se retrouvera à des positions différentes.

Tout d'abord, on regarde les moyennes de chaque question en fonction de l'ordre dans le questionnaire. On effectue la moyenne sur plusieurs années afin de compen-

ser les différences de difficulté de chaque question. L'évolution de la moyenne est représentée figure 6.6 page 6.6.

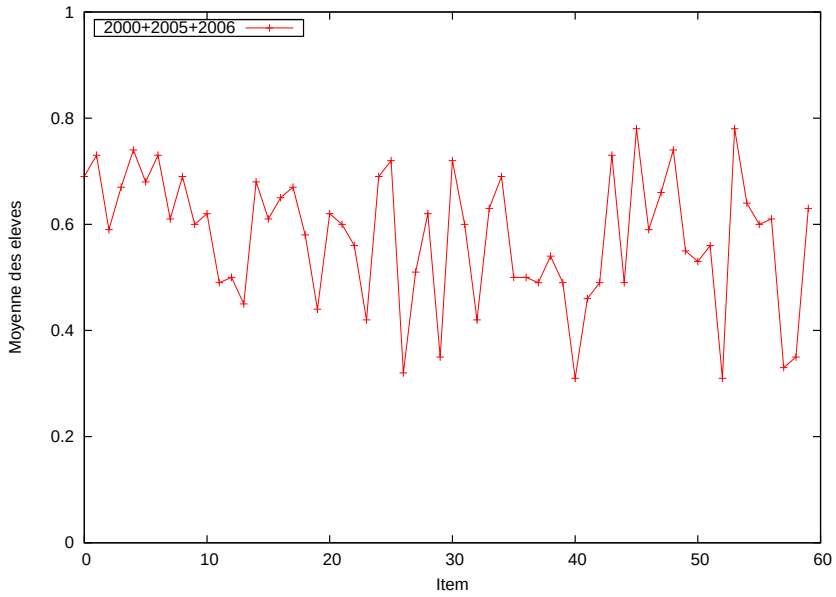
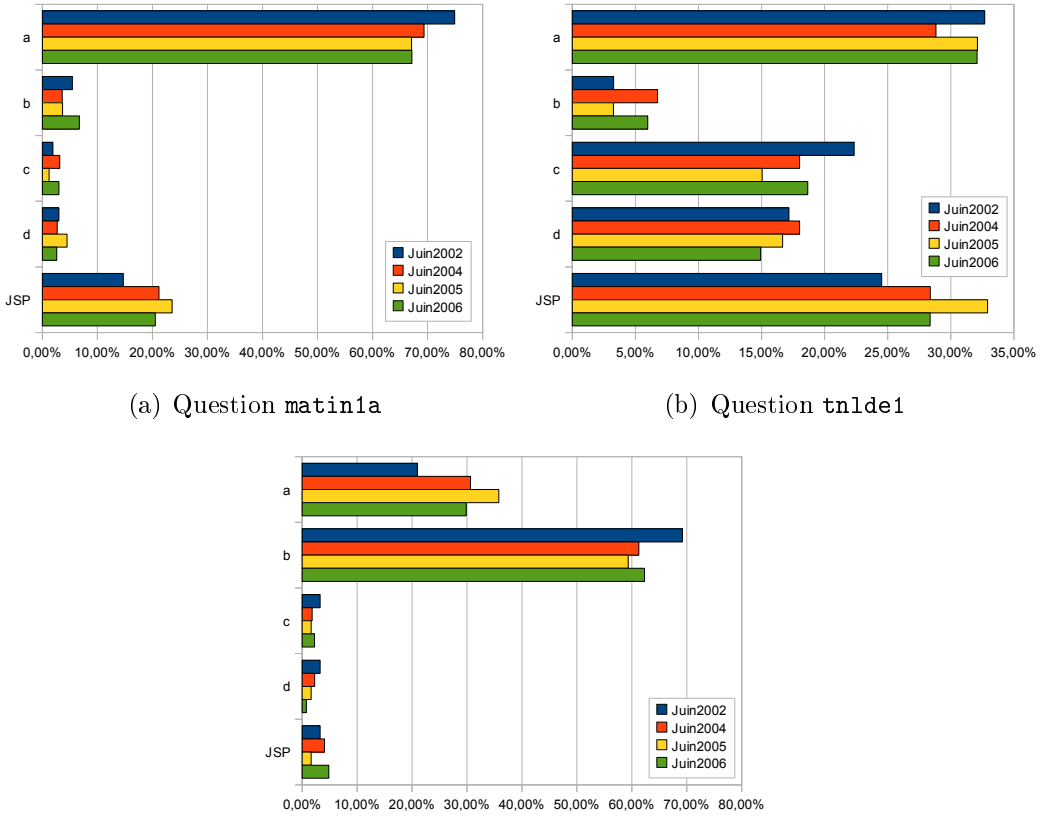


FIGURE 6.6 Évolution de la moyenne en fonction de la position dans le questionnaire

La moyenne ne présente pas de baisse significative pour les dernières questions. Cela semble indiquer que l'ordre des questions n'introduit pas particulièrement de biais. D'un autre côté, cela est compréhensible, car l'ordre des questions n'est pas le même pour tous les élèves lors d'une même session d'examen. En effet, il existe pour chaque session deux exemplaires de chaque questionnaire. Les questions finales sont donc soit la question 50, soit la question 60. On n'observe d'ailleurs pas non plus de baisse avant la 50^{e} question.

Pour confirmer cette conclusion, à savoir que l'ordre n'introduit pas de biais, nous avons pris soin de comparer les résultats donnés par les élèves à des questions qui reviennent régulièrement. La figure 6.7 montre la comparaison de plusieurs questions entre plusieurs années (JSP=Je ne sais pas).

Chaque question a été sélectionnée car elle a une position différente pour chaque



(a) Question `matin1a`

(b) Question `tnlde1`

(c) Question `dinde1e`

FIGURE 6.7 Résultats donnés à 3 questions différentes pour 4 années différentes

année. Le tableau 6.3 donne la position de la question en fonction de l'année.

TABLEAU 6.3 Position de la question dans le questionnaire

Question	Juin2002	Juin2004	Juin2005	Juin2006
`matin1a`	29e	40e	4e	26e
`tnlde1`	58e	60e	8e	53e
`dinde1e`	24e	3e	33e	3e

De plus, chaque question a un taux de réussite différent ce qui permet de s'assurer que ce n'est pas seulement parce que le taux de réussite est élevé que les réponses sont les mêmes.

Finalement, il apparaît de façon évidente que peu importe l'année, les réponses données aux questions sont les mêmes. Cela est d'autant plus frappant avec la question `tnlde1`, car on se rend compte que les erreurs commises sont les mêmes et dans les mêmes proportions et que la même proportion de personnes indiquait ne pas savoir chaque année.

On peut définitivement exclure l'influence de l'ordre sur les résultats obtenus lors de la classification.

6.5.2 Découpage en thèmes

Comme l'ordre ne semble pas influer sur les résultats de chaque question, il faut chercher une autre explication. Une explication réside dans le fait que le questionnaire est multidimensionnel : il fait appel à des habiletés et connaissances orthogonales et peu d'items portent sur une seule et même habileté ou connaissance, de la résolution de problème en géométrie, à la connaissance de formules d'intégration, par exemple. Il serait donc normal de n'avoir que peu de liens entre les items dans la mesure où ils mesurent des traits plus ou moins indépendants.

Pour vérifier cette hypothèse, nous avons donc décidé de découper le questionnaire

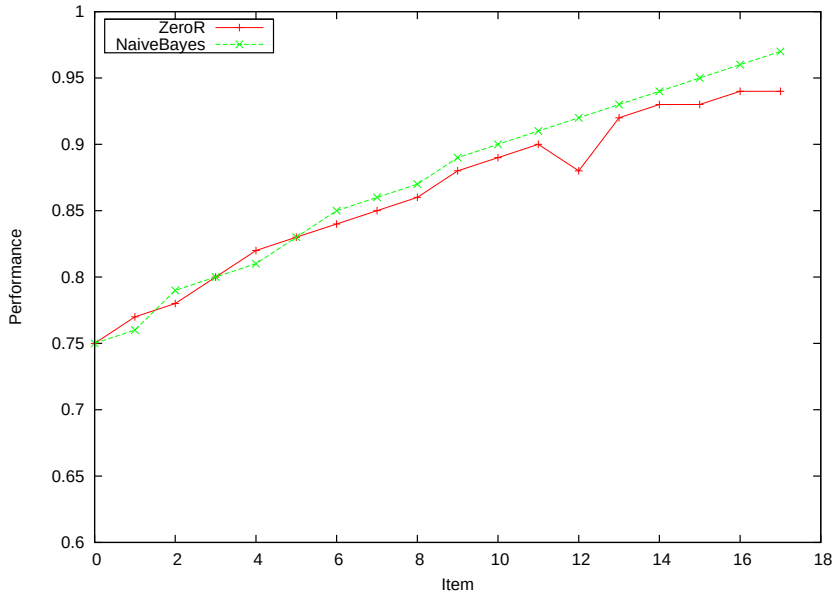
selon le thème indiqué pour chaque question. On se retrouve alors avec les groupes suivants :

- Calcul différentiel et intégral
- Élémentaire
- Géométrie et trigonométrie
- Vecteurs et matrices

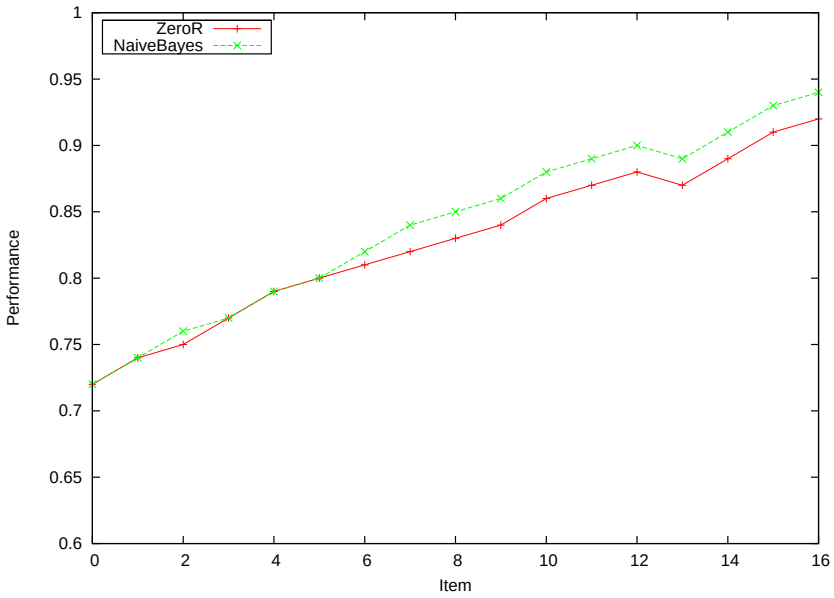
Nous avons donc découpé les questions en 4 groupes puis appliqué le banc d'essai à chaque groupe de questions. On se limite au classificateur sans inférence pour avoir une référence et au classificateur bayésien naïf qui donne les meilleurs résultats le plus rapidement possible. Les résultats obtenus pour les deux premiers groupes sont représentés sur la figure 6.8. Les résultats des autres groupes sont sensiblement identiques et ne sont donc pas reportés ici.

Malheureusement, les résultats que l'on obtient sont similaires à ceux que l'on avait auparavant, à savoir que le classificateur bayésien ne parvient pas à faire mieux que le classificateur sans inférence ou en tout cas pas de façon significative.

Finalement, nous nous sommes assuré que les données que nous possédons ne correspondent pas à des réponses aléatoires (sinon la figure 6.7 le montrerait). Il faudrait au final effectuer d'autres tests sur ces données pour voir si le problème est lié à une indépendance des questions entre elles ou si le problème vient d'ailleurs.



(a) Calcul différentiel et intégral



(b) Élémentaire

FIGURE 6.8 Performance en prenant en compte le découpage en thèmes

Chapitre 7

Applications client et serveur

Une fois arrivé à une conclusion dans le choix du classificateur, il est judicieux de mettre en place une structure qui permet d'effectuer des tests sans avoir à gérer la complexité inhérente au problème. Pour cela, nous avons choisi de développer un système client/serveur qui permettra aussi d'intégrer notre solution à des environnements d'apprentissage existants tel que ceux utilisés à l'École Polytechnique.

De plus, un tel découpage permettra de faire des mises à jour au niveau du serveur sans avoir à modifier en profondeur le client, et donc cela pourra être fait de manière transparente pour les utilisateurs.

Pour finir avec les avantages du découpage en client et serveur, il faut aussi évoquer les améliorations prévues pour le projet avec notamment l'ajout pour le guidage des ontologies des domaines. Cela permet de limiter le rôle de chaque serveur et de rendre les différentes opérations modulaires. La figure 7.1 permet de schématiser ce raisonnement.

Le schéma ne prend pas en compte un éventuel agent extérieur qui permettrait de choisir quel compétence ou connaissance serait la plus pertinente à évaluer ou à entraîner. C'est le rôle d'un éventuel guide d'apprentissage. Pour l'instant, on considère que le répondant choisit cela par lui-même.

Les différents éléments du schéma sont les suivants :

– **le serveur de questions**, qui contient le libellé de chaque question ; il peut s'agir tout simplement d'un serveur de fichier contenant les questions dans un

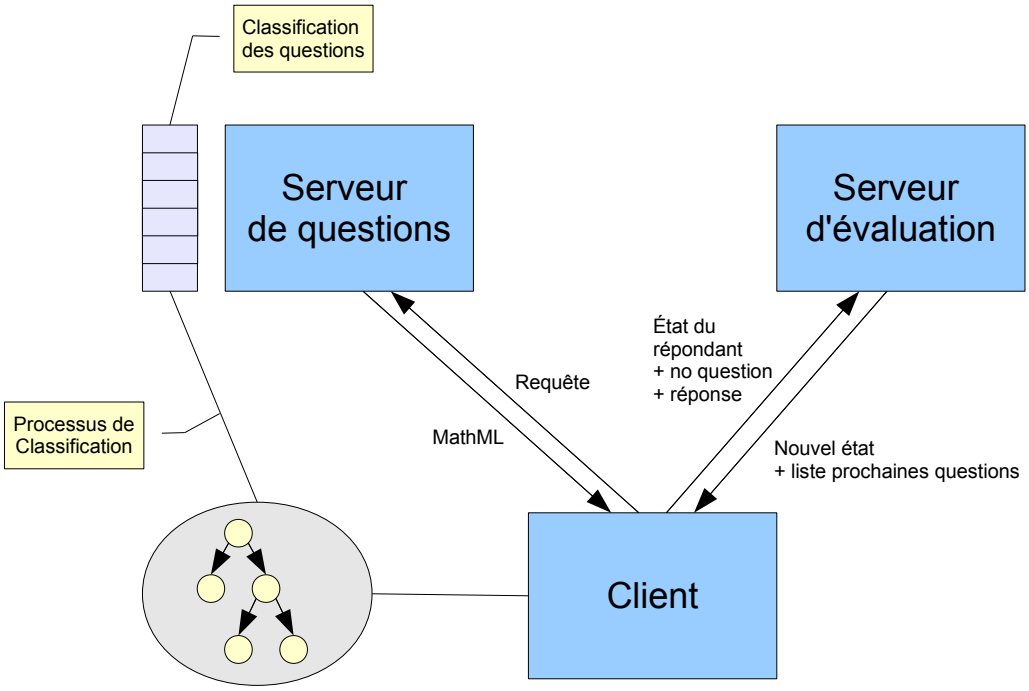


FIGURE 7.1 Architecture client-serveur choisie

format XML/MathML ;

- **le serveur d'évaluation**, qui évalue le répondant en fonction de l'état précédent et des nouvelles réponses ; il fournit également une liste ordonnée des questions en fonction de la pertinence de les poser ;
- **l'ontologie**, qui pourra être utilisée par le client pour choisir parmi la liste fournie par le serveur d'évaluation une question qui contient la compétence ou la connaissance que l'on cherche à évaluer en priorité ;
- **la classification des questions**, qui est effectuée en fonction de l'ontologie considérée et qui devrait pouvoir s'adapter si on modifie l'ontologie.

Une fois cette architecture déterminée, il faut définir les protocoles que l'on va mettre en place entre les différents éléments, puis déterminer le type de programmation qui sera utilisé pour chaque élément.

Tout d'abord, comme mentionné précédemment, le serveur de questions peut n'être qu'un serveur de fichiers ou un serveur web qui fournit les libellés des questions sous forme MathML. Cela permet d'avoir un système très simple dont la requête contiendra simplement l'identificateur de la question.

Ensuite, la communication entre le serveur d'évaluation et le client se fait en XML afin d'avoir une solution la plus simple possible. En effet, il existe de nombreux analyseurs syntaxiques XML et cela dans tous les langages utilisés couramment sur le Web. Le contenu des arbres XML ainsi échangés peut se retrouver en annexe B page 74.

De plus, on peut noter que nous avons pris parti de concevoir un serveur sans état, c'est-à-dire qui n'enregistre pas les sessions des différents clients. Ce sera au client de conserver et de retransmettre l'ensemble des données nécessaires à chaque communication. Cela permet entre autre de répartir la charge de travail sur plusieurs serveurs au besoin mais aussi de permettre au client de sauvegarder simplement la

session de travail sans ajouter de cas non fonctionnels au serveur.

Quant au type de serveur utilisé, trois contraintes majeures influencent le choix du langage de programmation :

- il doit être possible de déchiffrer facilement les arbres XML transmis ;
- le code fourni doit pouvoir s'exécuter le plus facilement possible à la fois sous un environnement Unix/Linux et sous un environnement Microsoft Windows ;
- il faut choisir un langage pour lequel l'implémentation des différents classificateurs est déjà effectuée ou à tout le moins aisée, comme par exemple R pour *POKS* ou Java pour les classificateurs du banc d'essai *WEKA*.

Ces trois contraintes d'adaptabilité ont orienté notre choix vers un serveur sur la plate-forme Tomcat. La compatibilité avec Java est immédiate et il existe des librairies de communication avec R qui nous permettent de gagner du temps de développement au cas où notre choix final se porte sur un classificateur développé en R. De plus, il existe des logiciels qui permettent d'exécuter un code Tomcat aussi bien sous Linux que sous Windows et il existe naturellement de nombreux analyseurs syntaxiques XML de qualité implémentés en Java.

Chapitre 8

Conclusion

Nous allons revenir ici sur ce qui a été effectué durant ce projet. Nous présenterons ensuite les limites de l'étude et enfin nous donnerons des pistes d'explorations et d'amélioration pour de futures recherches dans le domaine.

8.1 Synthèse des travaux

Nous avons effectué tout d'abord un survol du domaine composé par l'apprentissage assisté par ordinateur et par l'évaluation automatique d'un répondant. Nous avons par la suite explicité les techniques de fouille de données existantes et notamment celles qui sont habituellement utilisées dans le domaine de l'évaluation automatique et adaptative.

Nous avons donc pu noter que l'évaluation adaptative permet de réduire le temps d'évaluation et d'effectuer un diagnostic plus précis des connaissances d'un élève. De plus, on aura pu remarquer que les techniques utilisées sont souvent basées sur des réseaux bayésiens et qu'une réelle comparaison des différents classificateurs n'a pas été faite dans le domaine des questionnaires adaptatifs. Notre décision d'effectuer un comparatif des différents classificateurs est donc motivée et nous permettra d'effectuer un choix éclairé afin de réaliser le meilleur système possible pour le guide d'apprentissage projeté.

Nous avons alors sélectionné un nombre réduit de classificateurs qui ont pour parti-

cularité d'être soit très simples (comme le classificateur sans inférence) et permettant d'avoir une base de comparaison meilleure que le hasard, soit basés sur des théories plus complexes mais promettant de meilleures performances, selon les différents articles de la littérature que nous avons consultés.

Une fois cela fait, il fallait se procurer un ensemble de jeux de données sur lesquelles tester notre ensemble de classificateurs. Pour cela, outre 3 jeux de données déjà disponibles, nous avons pu disposer des archives de l'examen facultatif de mathématiques proposé aux nouveaux arrivants du baccalauréat.

Nous avons alors réalisé trois types d'expérimentation : une première évaluation qui permet de vérifier que les classificateurs choisis sont de bon niveau, une seconde expérimentation avec un ordre de questions aléatoire qui nous permettait d'éviter le problème de l'identification de l'ordre optimal dans lequel les questions allaient devoir être posées et enfin une dernière expérimentation avec un ordre des questions optimisé en fonction de l'entropie a priori de chaque question.

Cela nous a permis de vérifier que le classificateur bayésien naïf était bien souvent aussi efficace que le classificateur en réseau *POKS* et cela avec un temps de création du modèle beaucoup plus faible. Cependant la différence entre les deux classificateurs est dans la plupart des cas faible. Par contre, nous avons pu voir que les réseaux bayésiens, bien que performants pour un nombre faible de questions, fournissent de mauvais résultats lorsque le nombre de questions devient trop grand (typiquement pour un nombre de question supérieur à la cinquantaine). Enfin, les arbres de décision s'avèrent également être moins efficaces que les deux premiers classificateurs cités.

Ces tests ont aussi mis en évidence un comportement étrange avec les jeux de données extraits des tests de mathématiques. En effet, pour ces derniers on observe des résultats presque similaires pour l'ensemble des classificateurs, le classificateur sans inférence inclus. On a donc effectué des tests supplémentaires avec ces données,

notamment en essayant de séparer les différents thèmes identifiés, ou encore en vérifiant que les résultats obtenus pour les différentes questions étaient les mêmes lorsque celles-ci étaient placées ailleurs dans le questionnaire papier.

Enfin, nous avons développé parallèlement à cela des applications client–serveur qui permettent de mettre en application les différents classificateurs dans un cadre réel simplifié. Les deux applications communiquent via un protocole en XML documenté et permettent d'inclure l'application dans un grand nombre de plate-formes préexistantes.

8.2 Limitations des travaux

Tout d'abord, on peut noter que l'objectif initial, qui était de sélectionner un classificateur efficace dans le cadre d'un questionnaire adaptatif pour l'École Polytechnique, n'a pas été atteint. En effet, les résultats démontrent que les classificateurs étudiés ne sont pas plus performants qu'une approche basée sur des tests classiques, non adaptatifs.

Ensuite, nous n'utilisons pas pour l'instant les classifications des questions et les relations possibles avec l'ontologie existante. Ce rapprochement nous permettrait peut-être de découvrir que d'autres classificateurs sont plus efficaces que ceux qui ont été identifiés.

Enfin, il faudrait peut-être choisir des classificateurs moins classiques, qui sont possiblement plus compliqués à mettre en place, mais qui permettrait peut-être d'obtenir de meilleurs résultats.

8.3 Perspectives et recherches futures

Nos travaux permettent de conclure sur la performance des classificateur *POKS* et bayésiens naïfs dans certains cas. Cependant on a vu que certains jeux de données faisaient échec à ces classificateurs. Il faudrait donc déterminer les raisons qui font que les données recueillies lors des tests de mathématiques posent problèmes. Pour cela, il faudrait éventuellement construire un nouveau questionnaire qui suivrait des critères à déterminer. Le travail demandé serait alors d'identifier les raisons qui font que certains jeux de données permettent d'avoir d'excellents résultats (difficulté des questions par exemple) et de construire un questionnaire en fonction de ces critères.

De même, nous n'avons pas testé différentes méthodes de sélection pour la prochaine question car celle qui a été retenue semblait suffisamment performante pour effectuer une comparaison éclairée. Cela dit, cela pourrait être intéressant de se pencher sur la question.

D'autre part, notre expérience ne prend pas en compte les compétences et les connaissances à mettre en $\frac{1}{2}$uvre pour répondre à une question. Il faudrait donc s'interroger sur les moyens disponibles pour prendre cela en compte, notamment au niveau de la sélection de la prochaine question dans le cadre d'un guide d'apprentissage complet.

Et pour finir, il faut tout simplement poursuivre le travail de création du guide d'apprentissage, d'un point de vue programmation de l'interface d'abord mais aussi pour ce qui est d'utiliser au mieux les évaluations précises que l'on possède des élèves.

Références

AINSWORTH, S. et FLEMING, P. (2005). Evaluating a mixed-initiative authoring environment : Is redeem for real ? *Proceeding of the 2005 conference on Artificial Intelligence in Education*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 9–16.

ALLEN, M. J. et YEN, W. (2001). *Introduction to Measurement Theory*. Waveland Press.

ANASTASI, A. (1994). *Introduction Ã la psychomÃ©trie*. GuÃ©rin Ã©diteur.

AROYO, L. et DICHEVA, D. (2004). The new challenges for e-learning : The educational semantic web. *Educational Technology & Society*, 7, 59–69.

BRUSILOVSKY, P., EKLUND, J. et SCHWARZ, E. (1998). Web-based education for all : A tool for development adaptive courseware. *Computer Networks and ISDN Systems*. 1–7.

BRUSILOVSKY, P., SCHWARZ, E. et WEBER, G. (1996). Elm-art : An intelligent tutoring system on world wide web. Springer Verlag, 261–269.

CEDERBERG, S. et WIDDOWS, D. (2003). Using lsa and noun coordination information to improve the precision and recall of automatic hyponymy extraction. *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. Association for Computational Linguistics, Morristown, NJ, USA, 111–118.

CIMIANO, P. (2006). *Ontology Learning and Population from Text : Algorithms, Evaluation and Applications*. Springer.

COOPER, G. F. et DIETTERICH, T. (1992). A bayesian method for the induction of probabilistic networks from data, 309–347.

D., B. (2005). *Measuring the Mind : Conceptual Issues in Modern Psychometrics*. Cambridge University Press.

DESMARAIS, M. et GAGNON, M. (2006). Bayesian student models based on item to item knowledge structures. *EC-TEL*. 111–124.

DESMARAIS, M., MALUF, D. A. et LIU, J. (1995). User-expertise modeling with empirically derived probabilistic implication networks. *User Model. User-Adapt. Interact.*, 5, 283–315.

DESMARAIS, M. C. et LIU, J. (1993). Exploring the applications of user-expertise assessment for intelligent interfaces. S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel et T. White, éditeurs, *Proceedings of the Conference on Human Factors in computing systems, INTERCHI'93*. ACM Press, New York, 308–313.

DESMARAIS, M. C., MESHKINFAM, P. et GAGNON, M. (2006a). Learned student models with item to item knowledge structures. *User Modeling and User-Adapted Interaction*, 16, 403–434.

DESMARAIS, M. C., MESHKINFAM, P. et GAGNON, M. (2006b). Learned student models with item to item knowledge structures. *User Modeling and User-Adapted Interaction*, 16, 403–434.

DESMARAIS, M. C., VILLARREAL, A. et GAGNON, M. (2008). Adaptive test design with a naive bayes framework. R. S. J. de Baker, T. Barnes et J. E. Beck, éditeurs, *EDM*. www.educationaldatamining.org, 48–56.

DOIGNON, J.-P. et FALMAGNE, J.-C. (1999). *Knowledge Spaces*. Springer-Verlag, Berlin.

DOMINGOS, P. et PAZZANI, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29, 103–130.

HALD, A. (1999). On the history of maximum likelihood in relation to inverse probability and least squares. *Statist. Sci.*, 14, 214–222.

HAMBLETON, R. K., SWAMINATHAN, H. et ROGERS, H. J. (1991). *Fundamentals of Item Response Theory*. Sage Press.

HAMBLETON, R. K., SWAMINATHAN, H. et ROGERS, H. J. (2001). *The Basics of Item Response Theory*. ERIC Clearinghouse on Assessment and Evaluation, University of Maryland, College Park, MD.

HECKERMAN, D. (1995). A tutorial on learning with Bayesian networks. Rapport technique MSR-TR-95-06, Microsoft Research (MSR), Redmond, WA.

JAMESON, A. (1996). Numerical uncertainty management in user and student modeling : An overview of systems and issues.

KAMBOURI, M., KOPPEN, M., VILLANO, M. et FALMAGNE, J.-C. (1994). Knowledge assessment : tapping human expertise by the query routine. *Int. J.*

Hum.-Comput. Stud., 40, 119–151.

KOEDINGER, K., ANDERSON, J., HADLEY, W. et MARK, M. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8, 30–43.

KOEDINGER, K. R., ALEVEN, V., HEFFERNAN, N. T., MCLAREN, B. M. et HOCKENBERRY, M. (2004). Opening the door to non-programmers : Authoring intelligent tutor behavior by demonstration. *Intelligent Tutoring Systems*. 162–174.

LEWIS, C. et SHEEHAN, K. (1990). Using Bayesian decision theory to design a computerized mastery test. *Applied Psychological Measurement*, 14, 367–386.

LOOI, C.-K., MCCALLA, G. I., BREDEWEG, B. et BREUKER, J., éditeurs (2005). *Artificial Intelligence in Education - Supporting Learning through Intelligent and Socially Informed Technology, Proceedings of the 12th International Conference on Artificial Intelligence in Education, AIED 2005, July 18-22, 2005, Amsterdam, The Netherlands*, vol. 125 de *Frontiers in Artificial Intelligence and Applications*. IOS Press.

MARTIN, J. et VANLEHN, K. (1995). Student assessment using bayesian nets. *International Journal of Human-Computer Studies*, 42, 575–591.

MISLEVY, R. et CHANG, H. (2000). Does adaptive testing violate local independence ? *Psychometrika*, 65, 149–156.

NUZZO-JONES, G., WALONOSKI, J. A., HEFFERNAN, N. T. et LIVAK, T. (2005). The extensible tutor architecture : A new foundation for its. Looi *et al.*

(2005), 902–904, 902–904.

QUINLAN, R. J. (1993). *C4.5 : Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann.

RISH, I. (2005). An empirical study of the naive bayes classifier. *IJCAI-01 workshop on "Empirical Methods in AI"*.

RUDNER, L. M. (2002). An examination of decision-theory adaptive testing procedures. *Proceedings of American Educational Research Association*. New Orleans, 437–446.

SAVAGE, L. J. (1976). On rereading R. A. Fisher. *Ann. Statist.*, 4, 441–500. Edited posthumously by John W. Pratt, With a discussion by B. Efron, Churchill Eisenhart, Bruno de Finetti, D. A. S. Fraser, V. P. Godambe, I. J. Good, O. Kempthorne and Stephen M. Stigler.

THURSTONE, L. L. (1959). *The Measurement of Values*. The University of Chicago Press.

TURNER, T. E., MACASEK, M. A., NUZZO-JONES, G., HEFFERNAN, N. T. et KOEDINGER, K. R. (2005). The assistment builder : A rapid development tool for its. Looi *et al.* (2005), 929–931, 929–931.

VANLEHN, K., LYNCH, C., SCHULZE, K., SHAPIRO, J., SHELBY, R., TAYLOR, L., TREACY, D., WEINSTEIN, A. et WINTERSGILL, M. (2005). The andes physics tutoring system : Five years of evaluation. *Proceedings of AIED'05*. 678–685.

VANLEHN, K. et MARTIN, J. (1998). Evaluation of an assessment system based on bayesian student modeling. *INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE IN EDUCATION*, 8, 179–221.

VANLEHN, K. et NIU, Z. (2001). Bayesian student modeling, user interfaces and feedback : A sensitivity analysis. *International Journal of Artificial Intelligence in Education*, 12, 154–184.

VOMLEL, J. (2004). Bayesian networks in educational testing. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 12, 83–100.

WITTEN, I. H. et FRANK, E. (2005). *Data Mining : Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann.

WOOLF, B. P. et CUNNINGHAM, P. (1987). Building a community memory for intelligent tutoring systems. *AAAI*. 82–89.

ZOUAQ, A. et NKAMBOU, R. (2009). Evaluating the generation of domain ontologies in the knowledge puzzle project. *IEEE Transactions on Knowledge and Data Engineering*, 99.

Annexe A

Utilisation de WEKA à travers l'API Java

Afin de pouvoir tester rapidement un grand nombre de classificateurs, il existe des bancs d'essais complets qui regroupent un grand nombre de classificateurs classiques. Parmi ceux-ci, il existe le banc d'essais *WEKA* développé en Java par l'université de Waikato. Pour un usage courant, il présente l'avantage de posséder une interface graphique de qualité qui permet d'effectuer rapidement des comparaisons simples pour lesquelles seul le type de classificateur change. Cependant, si on veut effectuer un très grand nombre de tests avec beaucoup de paramètres qui varient, il est plus efficace d'utiliser l'API Java existante et de programmer une classe qui permettra d'effectuer les tests voulus.

Nous allons donc présenter ici des extraits du code produit avec les commentaires adéquats afin d'expliquer rapidement comment fonctionne cette API.

Listing A.1 Inclusion des librairies

```
1 import weka.core.*;       //Paquets de l'API weka
  import java.io.*;
3         //Paquets d'entree/sortie Java afin de charger les
              fichiers de donnees
```

Listing A.2 Chargement des données

```
FileReader reader = new FileReader(filename);
        //On charge le fichier de donnees grace aux outils java
Instances datas = new Instances(reader);
        //Et on insere le tout dans l'objet Instances defini par
           WEKA
```

Listing A.3 Getters utiles de l'objet Instances

```
int nbA = datas.numAttributes();
        //Nombre d'attributs d'une instance
int nbI = datas.numInstances();
        //Nombre d'instances d'un ensemble de donnees
Instance data = datas.instance(n);       //instance de rang n

Instances testDatas = datas.testCV(n,i);
Instances trainDatas = dats.trainCV(n,i);
        //Divise les donnees en n parties pour une validation
           croisee a n replis en extrayant la i-eme pour les tests
            et le reste pour l'entrainement
```

Listing A.4 Création des classificateurs

```
classificateur = new weka.classifiers.rules.ZeroR();
classificateur = new weka.classifiers.trees.J48();
classificateur = new weka.classifiers.bayes.NaiveBayes();
classificateur = new weka.classifiers.bayes.BayesNet();
        //Instanciation d'un classificateur (cf doc pour les
           autres classificateurs)
```

```
7 datas.setClassIndex(k);
        //Pour indiquer au classificateur quelle est la classe
9
  weka.classifiers.bayes.net.search.local.K2 cK2 = new weka.
     classifiers.bayes.net.search.local.K2();
11 cK2.setInitAsNaiveBayes(false);
  cK2.setMaxNrOfParents(3);
13 classificateur.setSearchAlgorithm(cK2);
        //Creation d'une instance pour utiliser le mode de
           recherche K2 pour le reseau bayesien avec les
           parametres choisis
15
  classificateur[i].buildClassifier(datas);
17      //Calcul du classificateur a partir des donnees datas
```

Listing A.5 Ordre des questions

```
1 int nbA = datas.numAttributes();
  indiceQuestion = (int) Math.round((nbA-1)*Math.random());
3       //Choix aleatoire

5 proba = classificateur.distributionForInstance(reponsesJusqueLa);
        //Probabilites de chaque possibilites (0 ou 1)
7 entro = -proba[0]*Math.log(proba[0])-proba[1]*Math.log(proba[1]);
        //Calcul de l'entropie pour chaque question
```

Listing A.6 Inférence et comparaison

```
datas.setClassIndex(k);
```

```
2 inference = classificateur.classifyInstance(datas.instance(i));
  reelle = datas.instance(i).value(k);
4         //On veut la valeur devinee et la valeure reelle pour le i
            -eme repondant a la question k
```

Pour plus d'information, se référer au chapitre 2.13 de Witten et Frank (2005) et à la documentation en ligne à `http://weka.sourceforge.net/doc/`.

Annexe B

Applications client/serveur

Dans cette section, on présente les fichiers XML échangés entre le client et le serveur et on explique l'utilité de chaque élément. Il faut garder à l'esprit que le serveur ne garde pas de sessions et que l'ensemble des échange est *sans état*.

En premier lieu, voici l'arbre envoyé par le client au serveur. Afin de simplifier la compréhension de l'échange, on présentera ici l'état des arbres XML dans 3 cas :

1. avant de commencer le test ;
2. avant de poser la première question, après le choix du questionnaire ;
3. le cas courant.

Client → Serveur : Initialisation, début du test

Dans ce cas, il suffit d'envoyer un arbre vide au serveur qui en déduira qu'il s'agit du début du test. Il renverra alors la liste des questionnaires disponibles ainsi que la liste des méthodes de sélection de la prochaine question.

Listing B.1 Le début du test

```
<?xml version="1.0"?>
<ClientState/>
```

Client → Serveur : Après le choix du questionnaire

Le choix du questionnaire et de la méthode n'a pas besoin d'être précisé au serveur dans un échange à part car le serveur ne garde de toute façon pas trace de l'échange.

On est maintenant avant la première question et il s'agit de déterminer quel est l'indice de cette dernière. Pour cela, on envoie au serveur le nom du questionnaire, la méthode de sélection de la prochaine question et on indique que l'on a encore répondu à aucune question.

Listing B.2 Avant la première question

```
<?xml version="1.0"?>
<ClientState>
        <currentQuestionnaire>
                nom_du_questionnaire
        </currentQuestionnaire>
        <currentMethod> nom_de_la_methode </currentMethod>
        <lastItem>
                <index>-1</index>
                        <!-- -1 indique qu'il n'y en a pas -->
                <value>0</value>
                        <!-- 1 si juste, 0 si faux -->
        </lastItem
</ClientState>
```

Client → Serveur : État courant

Par la suite, le client doit renvoyer au serveur l'état présent. En fonction du classificateur choisi, on aura soit la solution de renvoyer l'historique des bonnes et des

mauvaises réponses ou plutôt les probabilités de chaque question. Cet historique est en fait ce que le serveur a envoyé après la question précédente et ne doit pas être modifié par le client.

Listing B.3 État courant

```
<?xml version="1.0"?>
<ClientState>
        <currentQuestionnaire>
                nom_du_questionnaire
        </currentQuestionnaire>
        <currentMethod> nom_de_la_methode </currentMethod>
        <lastItem>
                <index> n </index>
                        <!-- indice de la derniere question
                            repondue -->
                <value> 1 ou 0 </value>
                        <!-- 1 si juste, 0 si faux -->
        </lastItem
        <State>
                <item> x1 </item>
                <item> x2 </item>
                        <!-- il y a autant d items que de
                            questions. -->
                        <!-- x peut etre soit la reponse donnee a
                            la question correspondante ou alors la
                            probabilite de bonne reponse -->
        </State>
</ClientState>
```

On va maintenant s'intéresser à la réponse du serveur dans tous les cas. En résumé, il va renvoyer la liste des méthodes et des questionnaires après l'initialisation du client et il va renvoyer la liste ordonnée des questions et l'état actualisé après chaque question répondue par le client.

Serveur → Client : Initialisation

Lorsque le serveur reçoit de la part du client un arbre vide, il va renvoyer la liste des méthodes de sélection de la prochaine question qu'il connaît ainsi que la liste des questionnaires dont il a soit un modèle déjà construit soit les archives à partir desquelles il saura recréer un modèle. Il renvoie alors ces deux listes au client.

Listing B.4 Avant la première question

```
<?xml version="1.0"?>
<ServerState>
        <listQuestionnaire>
                <questionnaire> questionnaire1 </questionnaire>
                <questionnaire> questionnaire2 </questionnaire>
        </listQuestionnaire>
        <listMethod>
                <method> methode1 </method>
                <method> methode2 </method>
        </listMethod>
</ServerState>
```

Serveur → Client : État courant

Lorsque le client envoie la réponse à une nouvelle question, le serveur va calculer le nouvel état du modèle ainsi que l'ordre préférentiel des questions. Dans le cas où le client indique qu'il n'a encore répondu à aucune question, le serveur renvoie l'état initial du modèle.

Listing B.5 État courant

```
<?xml version="1.0"?>
<ServerState>
        <currentQuestionnaire>
                nom_du_questionnaire
\\      <currentMethod> nom_de_la_methode </currentMethod>
        <State>
                <item> x1 </item>
                <item> x2 </item>
                        <!-- il y a autant d items que de
                            questions. -->
                        <!-- x peut etre soit la reponse donnee a
                            la question correspondante ou alors la
                            probabilite de bonne reponse -->
        </State>
        <listItem>
                <item> n1 </item>
                <item> n2 </item>
                        <!-- cette liste est ordonnee et
                            correspond a l indice des questions a
                            poser -->
```

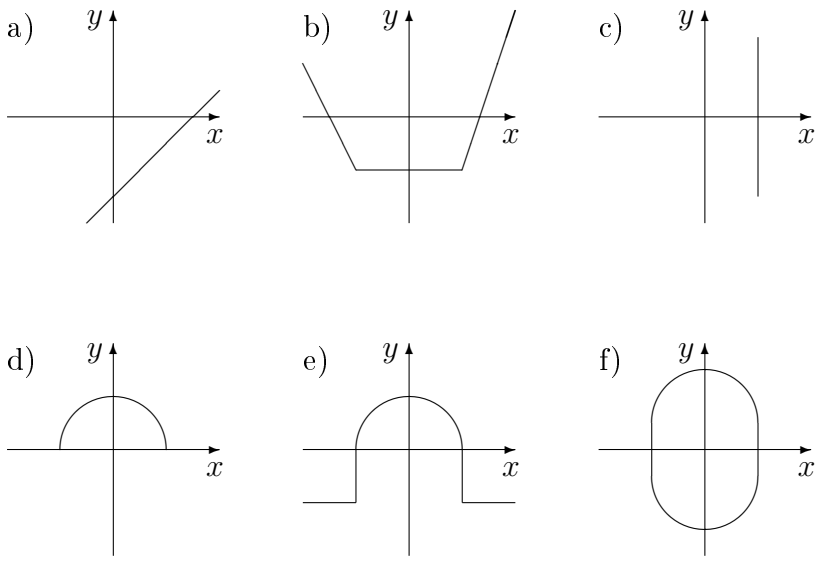
```
            </listItem>
17 </ServerState>
```

Annexe C

Exemples de questions

Voici quelques questions telles que posées lors d'un questionnaire typique de mathématiques.

1. Soit les graphes suivants



Les graphes représentant des fonctions sont

1) seulement (a),(b) et (d)

2) seulement (a),(b),(c) et (d)

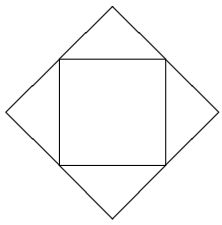
3) tous

4) tous sauf (c) et (f)

5) je ne sais pas

2. Si l'aire du carré intérieur est de 6 cm^2, l'aire du carré extérieur est de

   1) 9 cm^2
   2) 12 cm^2
   3) 24 cm^2
   4) 36 cm^2
   5) je ne sais pas



3. Soit la fonction

$$g(x) = \begin{cases} 1 & \text{si } x = 0 \\ \\ \dfrac{1}{x} & \text{si } x \neq 0 \end{cases}.$$

   Alors

   1) $g(x)$ est continue partout
   2) $g(x)$ est continue partout sauf en $x = 0$
   3) $g(x)$ est continue partout sauf en $x = 1$
   4) $g(x)$ est croissante pour $x > 0$
   5) je ne sais pas

4. Soit $M(2, \pi/3)$ un point en coordonnées polaires. Soit x et y les coordonnées cartésiennes de M. Alors la distance de M à l'origine est

   1) $x^2 + y^2$
   2) 2
   3) $2 \cos \pi/3$
   4) 4
   5) je ne sais pas

5. L'équation de la tangente au cercle d'équation $(x-1)^2+(y-2)^2=4$ passant par le point $A(3,2)$ est

   1) $x=2$
   2) $y=3$
   3) $2x-3y=0$
   4) autre chose
   5) je ne sais pas

6. Si $\overrightarrow{bc}=x\ \overrightarrow{ab}+y\ \overrightarrow{mb}$ alors

   1) $x=1$ et $y=-1$
   2) $x=2$ et $y=-1$
   3) $x=1$ et $y=-2$
   4) $x=2$ et $y=-2$
   5) je ne sais pas