Analyse des données évolutives : application aux données d'usage du Web

Alzennyr Gomes da Silva

Université Paris IX Dauphine
École Doctorale Décision, Informatique, Mathématiques et Organisation (EDDIMO)

N° attribué par la bibliothèque

Analyse des données évolutives : application aux données d'usage du Web

THÈSE

pour l'obtention du grade de

Docteur en Informatique
(Arrêté du 7 août 2006)

présentée et soutenue par

Alzennyr GOMES DA SILVA

le 24 septembre 2009 devant le jury composé de :

M. Edwin DIDAY, Professeur à l'Université Paris IX Dauphine	**Directeur de thèse**
M. Yves LECHEVALLIER, Directeur de Recherches à l'INRIA	**Co-directeur de thèse**
M. Gilbert SAPORTA, Professeur au CNAM-Paris	**Rapporteur**
M. Georges HEBRAIL, Professeur à l'ENST-Paris	**Rapporteur**
M. André HARDY, Professeur aux FUNDP-Namur	**Suffragant**
M. Witold LITWIN, Professeur à l'Université Paris IX Dauphine	**Suffragant**
M. Djamel ZIGHED, Professeur à l'Université Lumière Lyon 2	**Suffragant**
M. Francisco DE CARVALHO, Professeur à l'UFPE-Recife	**Suffragant**

Thèse préparée au sein du projet AxIS (INRIA, centre de Paris-Rocquencourt)

L'université n'entend donner aucune approbation ni improbation aux opinions émises dans les thèses : ces opinons doivent être considérées comme propres à leurs auteurs.

Résumé

Le nombre d'accès aux pages Web ne cesse de croître. Le Web est devenu l'une des plates-formes les plus répandues pour la diffusion et la recherche d'information. Par conséquence, beaucoup d'opérateurs de sites Web sont incités à analyser l'usage de leurs sites afin d'améliorer leur réponse vis-à-vis des attentes des internautes. Or, la manière dont un site Web est visité peut changer en fonction de divers facteurs. Les modèles d'usage doivent ainsi être mis à jour continuellement afin de refléter fidèlement le comportement des visiteurs. Ceci reste difficile quand la dimension temporelle est négligée ou simplement introduite comme un attribut numérique additionnel dans la description des données. C'est précisément sur cet aspect que se focalise la présente thèse. Pour pallier le problème d'acquisition des données réelles d'usage, nous proposons une méthodologie pour la génération automatique des données artificielles permettant la simulation des changements. Guidés par les pistes nées des analyses exploratoires, nous proposons une nouvelle approche basée sur des fenêtres non recouvrantes pour la détection et le suivi des changements sur des données évolutives. Cette approche caractérise le type de changement subi par les groupes de comportement (apparition, disparition, fusion, scission) et applique deux indices de validation basés sur l'extension de la classification pour mesurer le niveau des changements repérés à chaque pas de temps. Notre approche est totalement indépendante de la méthode de classification et peut être appliquée sur différents types de données autres que les données d'usage. Des expérimentations sur des données artificielles ainsi que sur des données réelles issues de différents domaines (académique, tourisme et *marketing*) ont été réalisées pour l'évaluer l'efficacité de l'approche proposée.

Mots-clés : Analyse de données, classification non supervisée, données évolutives, fouille d'usage du Web.

Abstract

Nowadays, more and more organizations are becoming reliant on the Internet. The Web has become one of the most widespread platforms for information change and retrieval. The growing number of traces left behind user transactions (e.g. : customer purchases, user sessions, etc.) automatically increases the importance of usage data analysis. Indeed, the way in which a web site is visited can change over time. These changes can be related to some temporal factors (day of the week, seasonality, periods of special offer, etc.). By consequence, the usage models must be continuously updated in order to reflect the current behaviour of the visitors. Such a task remains difficult when the temporal dimension is ignored or simply introduced into the data description as a numeric attribute. It is precisely on this challenge that the present thesis is focused. In order to deal with the problem of acquisition of real usage data, we propose a methodology for the automatic generation of artificial usage data over which one can control the occurrence of changes and thus, analyse the efficiency of a change detection system. Guided by tracks born of some exploratory analyzes, we propose a tilted window approach for detecting and following-up changes on evolving usage data. In order measure the level of changes, this approach applies two external evaluation indices based on the clustering extension. The proposed approach also characterizes the changes undergone by the usage groups (e.g. appearance, disappearance, fusion and split) at each timestamp. Moreover, the refereed approach is totally independent of the clustering method used and is able to manage different kinds of data other than usage data. The effectiveness of this approach is evaluated on artificial data sets of different degrees of complexity and also on real data sets from different domains (academic, tourism and marketing).

Keywords : Clustering, data analysis, evolving data, Web Usage Mining (WUM).

Remerciements

Je tiens à remercier en premier lieu tous les membres du jury de m'avoir fait l'honneur de leur participation à ma soutenance de thèse.

Je remercie Monsieur Gilbert SAPORTA et Monsieur Georges HEBRAIL d'avoir accepté de rapporter cette thèse et d'avoir pris le temps de lire ce manuscrit. Je vous remercie pour l'intérêt que vous avez porté à ce travail et pour vos remarques pertinentes qui m'ont permises d'améliorer considérablement la qualité de ce mémoire. Je remercie également Messieurs André HARDY, Witold LITWIN, Djamel ZIGHED et Francisco DE CARVALHO de m'avoir fait l'honneur d'être présents à ce jury de thèse ainsi que d'avoir examiné ce travail à travers de regards critiques et avisés.

Je tiens à remercier vivement mon directeur de thèse Monsieur Edwin DIDAY de m'avoir accepté en tant qu'étudiante sous sa direction à l'Université Paris-Dauphine. Je remercie également Madame Brigitte TROUSSE et Monsieur Yves LECHEVALLIER de m'avoir accepté au sein du projet AxIS. Un grand merci à vous trois pour la confiance que vous m'avez accordée, notamment lors de mon arrivée en France, époque à laquelle je prononçais à peine quelques mots de français. Soyez assurés d'avoir ma plus sincère et profonde reconnaissance.

Je remercie sincèrement Monsieur Francisco DE CARVALHO de m'avoir présenté à l'équipe AxIS et surtout, d'être à l'origine des premiers contacts afin que cette thèse puisse avoir eu lieu.

Egalement, cette thèse n'aurait sans doute pu voir le jour sans le soutien de mon co-directeur de thèse Monsieur Yves LECHEVALLIER. Son apport scientifique, son expertise et son regard critique m'ont été de précieux éléments à l'aboutissement des travaux de recherche décrits dans ce manuscrit. Pour votre confiance, votre encouragement et votre disponibilité au cours de ces dernières années, je vous adresse ma profonde reconnaissance et mes remerciements les plus sincères.

Je remercie également Messieurs Fabrice ROSSI et Patrice BERTRAND pour leurs judicieux conseils qui m'ont aidé à mener à bien les travaux de cette thèse.

D'autres personnes m'ont encouragé à finir ce travail par des gestes d'amitié dont je suis très reconnaissante. A titre d'exemple, je citerai Mesdames Anne-Marie Vercoustre et Marie-Aude Aufaure pour les avis et le soutien encourageants. Je ne pourrais sans doute oublier mes chers collègues de laboratoire que j'ai eu le plaisir de côtoyer durant ces quelques années, à savoir Abdourahamane BALDE, Abdouroihamane ANLI, Zeina JRAD, Alice MARASCU, Calin GARBONI et Malika CHARRAD. Un grand merci pour les échanges aussi bien culturels qu'académiques, sans oublier les moments de détente.

Je tiens aussi à mentionner le plaisir que j'ai eu à travailler dans une ambiance chaleureuse et détendue au sein du projet AxIS de l'INRIA-Rocquencourt, et j'en remercie ici tous les membres. Je remercie spécialement les assistantes Stéphanie AUBIN et Stéphanie CHAIX pour leur gentillesse, attention et prompte disposition envers tous les membres du projet.

Je remercie chaleureusement mes amies et compatriotes Patricia LIRA et Najla MATOS pour le partage de l'expérience en thèse qu'elles ont également vécu au même temps que moi.

Je remercie du fond du cœur ma famille, et toute spécialement ma mère, pour son encouragement et soutien inconditionnel, même à distance.

Je remercie également mon chéri pour sa compréhension et son soutien durant tous les moments, notamment ceux des longues relectures.

Enfin, pour tout et bien plus encore, je remercie Dieu.

Tous ceux qui viennent d'être cités ont eu chacun une importance capitale dans la réalisation de mes travaux de thèse. Je réalise qu'aujourd'hui ces rencontres m'ont apporté le soutien indispensable à l'aboutissement de cette somme de travail.

Table des matières

Liste des symboles

E	l'ensemble de données
n	taille de l'ensemble de données
p	dimension de l'espace de données
x_i	représentation vectoriel de l'i_{eme} individu de E, où $x_i = \{x_i^1, ..., x_i^p\}$
w_i	poids de l'individu i, avec $w_i \in [0, 1]$ et $\sum_{i=1}^{n} w_i = 1$
P	partition de l'ensemble de données E en K clusters, $P = (C_1, ..., C_l, ..., C_K)$
μ_l	poids du cluster l, où $\mu_l = \sum_{i \in C_l} w_i$ et $\sum_{l=1}^{K} \mu_l = 1$
α_l	effectif relatif du cluster C_l, où $\alpha_l \in [0, 1]$ et $\sum_{l=1}^{K} \alpha_l = 1$
g_l	centre de gravité du cluster l, où $g_l = \frac{1}{\mu_l} \sum_{i \in C_l} w_i x_i$
m	nombre de neurones constituant la carte de Kohonen
r_i	i_{eme} neurone de la carte de Kohonen
m_i	i_{eme} prototype en $\Re^p$ correspondant à l'i_{eme} neurone en $\Re^2$ de la carte de Kohonen
W^t	ensemble d'individus réunis dans une fenêtre t
G^t	ensemble de prototypes des clusters issus de la classification sur les individus de W^t
K^t	nombre de clusters obtenus par la méthode de classification sur les individus de W^t
P_1^t	partition obtenue par l'affectation des individus de W^t aux prototypes de G^{t-1}

P_2^t partition obtenue par la classification sur les individus de W^t

β_1 facteur de rétrécissement

β_2 facteur de grossissement

β_3 facteur de déplacement

$c^{'}$ classe clible des changements

Paramètres des algorithmes

$totalIndividuals$: nombre d'individus contenus dans une fenêtre de données artificielles

$maxWindow$: nombre maximum de fenêtres artificielles

$nbClicMIN$: nombre minimum de clics artificiels dans un thème de pages

$nbClicMAX$: nombre maximum de clics artificiels dans un thème de pages

$survivalThreshold$: seuil d'individus à être conservés par un cluster survivant

$splitThreshold$: seuil d'individus à être conservés par un cluster issu d'une scission

Introduction générale

Motivation et objectifs

En ce qui concerne les sources de données volumineuses et dynamiques, le Web est devenu l'exemple le plus pertinent grâce à l'augmentation colossale du nombre de documents mis en ligne et des nouvelles informations ajoutées chaque jour. Dans la perspective d'attirer de nouveaux clients et de répondre aux attentes des clients existants, un gérant de site Web bien avisé doit toujours garder à l'esprit que le fait d'offrir plus d'information ne constitue pas toujours une bonne solution. En réalité, les usagers d'un site Web apprécieront davantage la manière dont cette information est présentée au sein du site. L'analyse des traces d'usage (enregistrées dans les fichiers de type journaux par le serveur qui héberge le site Web) s'avère une pratique de plus en plus nécessaire pour mieux appréhender les pratiques des internautes. Dans ce contexte, la dimension temporelle joue un rôle très important car la distribution sous-jacente des données d'usage peut changer au cours du temps. Ce changement peut être provoqué par la mise à jour du contenu et/ou de la structure du site Web ou bien par le changement naturel d'intérêt des usagers d'un site Web.

Le changement de comportement des individus a également attiré l'attention de professionnels des sciences humaines. En effet, nous sommes actuellement dans la décennie du comportement (2000-2010) établie par l'Association Américaine de Psychologie (APA) et dont le but est de promouvoir des rencontres pour la conscientisation sur l'importance des recherches dans le domaine des sciences sociales et du comportement [1].

Les modèles d'accès à un site Web ont une nature dynamique et peuvent être influencés par certains facteurs temporels, comme par exemple : l'heure et le jour de la semaine où se déroule la visite d'un site Web, des événements saisonniers

1. Plus d'information sur l'adresse suivante : `http://www.decadeofbehavior.org/`

(vacances d'été, d'hiver, Noël), des événements ponctuels dans le monde (crises économiques, compétitions sportives, épidémies, etc.). La prise en compte de la dimension temporelle s'avère donc nécessaire pour l'analyse de ce type de données.

Durant les dernières années, toutes ces considérations ont motivé d'importants efforts dans l'analyse des traces des internautes ainsi que l'adaptation des méthodes de classification aux données du Web. Néanmoins, la plupart des méthodes consacrées à l'analyse des données d'usage prend en compte toute la période qui enregistre les traces d'usage. En conséquence, les modèles comportamentaux ressortis par ces méthodes sont ceux qui prédominent sur toute la période de temps analysée. Les comportements minoritaires passibles d'avoir lieu pendant de courtes périodes de temps restent ainsi inaperçus par les méthodes classiques. Dans le cadre du Web, quand un *webmaster* interroge les logs de son site, il souhaite que les résultats proposés en réponse à son interrogation soient fidèles à la période de temps analysée et non au comportement général remarqué tout au long de la période complète analysée. De plus, si une analyse des comportements des internautes ne réalise pas de suivi sur ces comportements au cours du temps, il serait impossible pour le *webmaster* de repérer la période de temps où le(s) possible(s) changement(s) de comportement d'usage ont eu lieu. Pour faire face à ce problème, une solution envisageable serait de définir une stratégie capable de fournir des moyens nécessaires pour que les responsables d'un site Web puissent être avertis lors de l'apparition, la disparition ou le changement des profils de comportement de leurs utilisateurs. Il serait également envisageable que l'administrateur du site puisse mesurer l'impact d'une nouvelle stratégie mise en ligne ainsi que la popularité des pages à l'aide de l'analyse des traces laissées par les internautes lors des visites.

De manière contradictoire à la quantité colossale des données mises en ligne sur Internet, l'une des difficultés les plus importantes liées à la fouille d'usage du Web est la pénurie de *benchmarks* de données d'usage pour l'application et la comparaison de différentes techniques d'analyse. Ceci est dû au fait que les données d'usage contiennent des informations de caractère privé.

Motivés par toutes ces considérations et afin de contribuer à l'avance de la recherche scientifique dans la communauté d'analyse des données, cette thèse s'attaque aux problématiques citées ci-dessus et présente deux groupes de contributions majeures. Premièrement, nous proposons une méthodologie pour la génération automatique de données artificielles d'usage sous la forme de tableau de contingence. Notre principal objectif est de rendre possible l'investigation de l'efficacité des dif-

férentes approches d'analyse de l'usage sur un ensemble de données contenant des changements établis au préalable. Notre méthodologie présente un algorithme générique de création de données artificielles et trois algorithmes spécialisés sur la simulation de changements liés à l'effectif et au déplacement des classes artificielles sur le temps. Tous les algorithmes sont délibérément présentés en détails afin de faciliter leur reproduction.

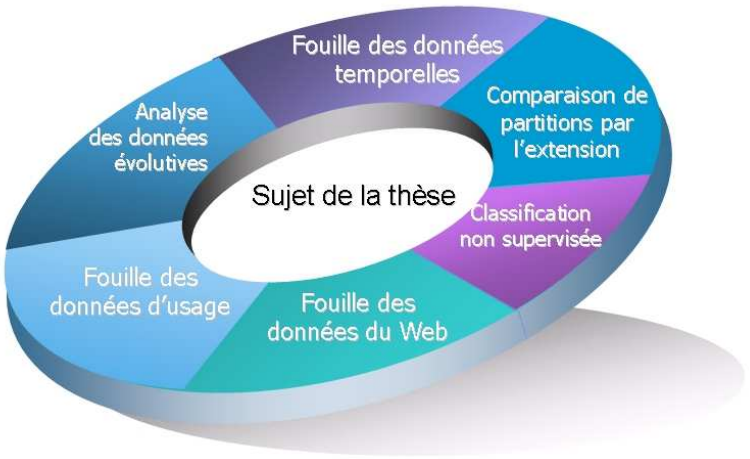

Figure 1 – Positionnement de la thèse.

La deuxième partie de nos recherches est axée sur l'investigation des stratégies de classification non supervisée les plus adaptées à la découverte des changements sur des données évolutives. Pour ce faire, nous réalisons initialement des analyses exploratoires sur différentes stratégies de classification non supervisée. A l'issue de cette étape d'investigation, nous proposons une nouvelle approche pour la découverte et le suivi des changements des données évolutives au cours du temps. L'approche proposée est totalement indépendante de la méthode de classification appliquée, ce qui implique que n'importe quelle méthode de classification fournissant un représentant (prototype) pour chaque cluster peut être rendue incrémentale par notre approche. Une autre caractéristique importante de l'approche proposée concerne la stratégie adoptée pour la détection des changements entre deux sous périodes consécutives de temps. Cette stratégie est basée sur l'extension, c'est-à-dire, sur l'ensemble d'individus ayant participé au processus de classification. De plus, notre approche fournie des fonctionnalités additionnelles permettant l'interprétation sémantique des changements vérifiés. Plus précisément, nous fournissons

quelques heuristiques permettant la vérification de la disparition d'un groupe de comportement, l'apparition d'un nouveau groupe de comportement, la fusion et la scission de différents groupes de comportement au cours du temps. Toutes ces considérations faites, le positionnement du présent travail se trouve à l'angle de différents courants de recherche illustrés dans la figure 1.

Organisation du mémoire

Ce document est composé de six chapitres, organisés de la façon suivante :

- **Le chapitre 1** présente le cadre général de nos travaux de recherche. Ils se placent plus globalement dans le domaine de la fouille des données et plus précisément dans le domaine de la fouille de données d'usage du Web. Ce chapitre présente également un état de l'art des travaux existants dans les domaines abordés ainsi que les enjeux majeurs y existants.
- **Le chapitre 2** présente les différentes méthodes de classification non supervisée exploitées dans la présente thèse, à savoir les méthodes hiérarchiques et les méthodes de partitionnement. En autre, ce chapitre passe en revue quelques critères proposés dans la littérature pour la détermination du nombre de classes dans une partition. Dans ce chapitre, nous présentons les critères de comparaison de partitions ainsi que les indices d'interprétation des classes adoptés dans notre approche de détection et suivi des changements sur des données évolutives.
- **Le chapitre 3** est consacré à la présentation détaillée des contributions de cette thèse. Celles-ci concernent la modélisation des données d'usage, la génération automatique de données artificielles d'usage, des analyses exploratoires de classification non supervisée, et finalement, la proposition d'une approche pour la détection et le suivi des changements sur des données évolutives.
- **Le chapitre 4** décrit la modélisation de notre application à l'aide du formalisme proposé par l'UML. Ce chapitre présente également les scripts SQL utilisés lors de l'extraction des variables descriptives des navigations.
- **Le chapitre 5** présente le cadre de nos expérimentations suivi par l'analyse des résultats obtenus à partir de l'application de notre approche sur différents jeux de données d'usage aussi bien artificielles que réelles.
- **Le chapitre 6** présente les apports des travaux de recherche décrits dans ce

document, et ce vis-à-vis de la communauté d'analyse de données et de l'utilisateur final. A la fin de ce chapitre, nous décrivons quelques pistes pouvant inspirer des futurs travaux dans la continuation de ceux présentés ici.

Chapitre 1

Fouille de données

1 Introduction

La fouille de données (Data Mining, en anglais) est définie comme l'application de l'analyse de données et des algorithmes de découverte sur les grandes bases de données ayant comme but la découverte de modèles non triviaux [46]. Plusieurs algorithmes ont été proposés afin de formaliser les nouveaux modèles découverts, de construire des modèles plus efficaces, de traiter de nouveaux types de données et de mesurer les différences entre les ensembles de données. Cependant, les algorithmes les plus traditionnels de la fouille de données supposent que les modèles soient statiques et ne tiennent pas compte de l'éventuelle évolution de ces modèles au cours du temps. Ces considérations ont motivé d'importants efforts dans l'analyse de données temporelles ainsi que l'adaptation des méthodes de la fouille de données statiques aux données qui évoluent avec le temps.

Le passage en revue des principaux volets de la fouille de données traités dans cette thèse constitue le corps de ce chapitre, suivi d'un état de l'art des travaux actuels dans ce domaine ainsi que d'une discussion sur les enjeux majeurs y existants.

2 Fouille de données temporelles

L'intérêt pour les bases de données temporelles a considérablement augmenté ces dernières années, par exemple dans les domaines de la finance, télécommunication, surveillance, etc. Un nombre de plus en plus important de prototypes et de systèmes sont mis en application afin de tenir compte de la dimension temporelle des données

de façon explicite pour, par exemple, étudier la variabilité au cours du temps des résultats des analyses.

La fouille de données temporelles (Temporal Data Mining (TDM), en anglais) [62] est une extension importante de la fouille de données classique car elle se centre plutôt sur l'analyse des activités plus que celle des états. De ce fait, elle permet de rechercher les associations de cause à effet en exploitant conjointement les proximités contextuelles et temporelles. Il s'agit d'exploiter le fait que les causes précèdent les effets, ce qui est difficile quand la dimension temporelle est négligée ou simplement introduite comme un attribut numérique additionnel dans la description des données.

En outre, la fouille de données temporelles est directement liée à la fouille de données sur des grands fichiers séquentiels. Par des données séquentielles, on entend les données qui sont ordonnées selon l'index de la séquence. Les données temporelles sont un cas particulier de données séquentielles dans lequel le temps joue le rôle d'indexation. Les séquences de gènes (ADN) et les séquences de mouvements dans un jeu d'échecs constituent d'autres exemples de données séquentielles. Ici, bien qu'il n'y ait aucune notion de temps en tant que tel, l'ordre des observations est très important et même indispensable pour la description et l'analyse de telles données.

Historiquement, le problème de la prévision de séries temporelles a été l'un des plus étudiés [19] dans le cadre de la météorologie, des finances et de la bourse. La principale différence entre la fouille de données temporelles et l'analyse de séries temporelles concerne la nature des informations que l'on veut estimer ou mettre en évidence. Le cadre de la fouille de données temporelles se prolonge au-delà des applications standards de prévision ou d'applications de contrôle pour l'analyse des séries temporelles. Dans l'analyse de séries temporelles la prévision joue un rôle central alors que dans fouille de données temporelles c'est plutôt l'évolution que l'on essaie de modéliser.

Dans [83], les auteurs résument les solutions proposées et les problèmes en suspens dans l'exploitation de données temporelles, au travers d'une discussion sur les règles temporelles et leur sémantique, mais aussi par l'investigation de la convergence entre la fouille de données et de la sémantique temporelle.

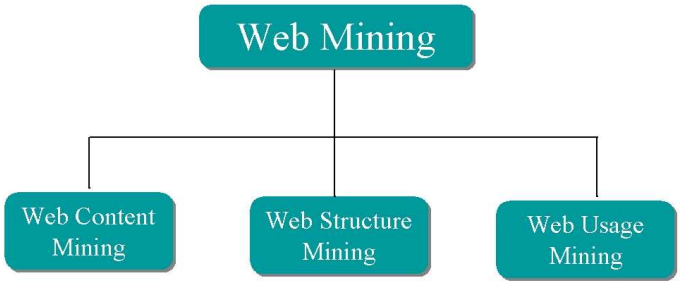

Figure 1.1 – Schéma classique de la fouille des données du Web.

3 Fouille de données du Web

La fouille du Web (Web Mining (WM), en anglais) [59] s'est développée à la fin des années 90 et consiste à utiliser l'ensemble des techniques de la fouille de données afin de développer des outils permettant l'extraction d'informations pertinentes à partir de données du Web (documents, traces d'interactions, structure des liens, etc.).

Selon l'objectif visé, plusieurs types d'études peuvent être réalisées (cf. figure 1.1), à savoir :

- L'analyse du contenu des pages Web (Web Content Mining) ;
- L'analyse des liens entre les pages Web (Web Structure Mining) ;
- L'analyse de l'usage des pages Web (Web Usage Mining).

Cette dernière branche du *Web Mining* consiste à analyser le comportement de l'utilisateur à travers l'analyse de son interaction avec le site Web. Cette analyse est notamment centrée sur l'ensemble des clics effectués par l'utilisateur lors d'une visite au site (on parle alors d'analyse du *clickstream*). L'intérêt est d'enrichir les sources de données utilisateur (bases de données clients, bases marketing, etc.) à partir des connaissances extraites des données brutes du *clickstream*, et ce afin d'affiner les profils utilisateur et les modèles comportementaux. C'est précisément sur cet axe, exploité plus en détail dans la section 5 du présent chapitre, que se focalise la présente thèse.

4 Fouille de données temporelles du Web

La fouille de données temporelles du Web (Temporal Web Mining (TWM), en anglais) est à l'intersection de la fouille du Web avec la fouille de données tempo-

relles. Selon [85], le TWM est le processus de découverte, d'extraction, d'analyse et de prévision de données contenant des informations temporelles, découvertes par l'application en temps réel des techniques de la fouille de données temporelles sur le Web. Selon [86], le TWM soutient l'aspect temporel des données du Web en considérant ces données comme des séries temporelles. Son but est de présenter la prévision comme une issue principale de la fouille du Web, spécifiquement de la fouille sur le contenu du Web (Web Content Mining). En résumé, la TWM proposée par [86] vise à faire des prévisions à partir du contenu des pages Web.

5 Fouille de données d'usage du Web

La fouille de données d'usage du Web (Web Usage Mining (WUM), en anglais) désigne l'ensemble de techniques basées sur la fouille de données pour analyser l'usage d'un site Web [91, 32, 89]. En d'autres termes, le WUM correspond au processus d'Extraction de Connaissances dans les Bases de Données (ECD) - ou Knowledge Discovery in Databases (KDD), en anglais - appliqué aux données d'usage du Web. Lorsqu'un utilisateur consulte un site Web, il navigue en effectuant une suite de *clics* avec sa souris et introduisant du texte avec son clavier. Ces informations déclenchent des requêtes (affichage d'une page du site, téléchargement d'un fichier, identification de l'utilisateur via un mot de passe, etc.) qui sont enregistrées en format texte et stockées de manière standardisée dans un fichier qui s'appelle *log Web*. Ce fichier est maintenu par le serveur HTTP qui héberge le(s) site(s) en question. Suivant la fréquentation du site, la taille du fichier log peut atteindre des proportions importantes, pouvant croître de quelques centaines de mégaoctets jusqu'à plusieurs dizaines de gigaoctets par mois.

A chaque page visitée, le serveur Web enregistre notamment dans le fichier log : l'adresse IP du client, la date et l'heure de la requête, la page consultée ou les fonctionnalités mises en oeuvre (téléchargement de fichier, clic sur image, etc.), le code de statut attribué à la requête (prend la valeur 200 en cas de réussite), le nombre d'octets transmis, la page précédemment visitée (ou le moteur de recherche utilisé pour rejoindre la page Web suivi des mots clés demandés), la configuration du client, c'est-à-dire, son navigateur Web (Firefox, Internet Explorer, etc.) et son système d'exploitation (Windows, Linux, Mac OS, etc.). Le format d'un fichier log (Extended Log File Format) a été standardisé par W3C [98]. Le tableau 1.1 illustre

un extrait de fichier log du serveur Web de l'INRIA.

adresse IP	date/heure	URL	statut	octets	page précédente	navigateur
194.78.232.8	[10/Sep/2001 :15 :33 :43 +0200]	GET /orion/liens.htm HTTP/1.1	200	1893	http ://www-sop.inria.fr/ orion/index.html	Mozilla/4.0 (compatible ; MSIE 5.0b1 ; Mac_PowerPC)
100.64.30.6	[10/Sep/2001 :15 :34 :07 +0200]	GET /stacs2002/home.html HTTP/1.0	200	483	http ://www.google.fr/ search ?hl=fr&q=inria +statistiques&meta=	Microsoft Internet Explorer 5.0
194.78.232.8	[10/Sep/2001 :15 :34 :09 +0200]	GET /orion/Telescope/Telesc.html HTTP/1.1	200	4433	http ://www-sop.inria.fr/ orion/liens.htm	Mozilla/4.0 (compatible ; MSIE 5.0b1 ; Mac_PowerPC)
100.64.30.6	[10/Sep/2001 :15 :34 :09 +0200]	GET /stacs2002/Images/affiche.jpg HTTP/1.0	200	281281	http ://www-sop.inria.fr /stacs2002/home.html	Microsoft Internet Explorer 5.0
194.78.232.8	[10/Sep/2001 :15 :34 :23 +0200]	GET orion/Telescope/Vsurv.html HTTP/1.1	200	2979	http ://wwwsop.inria.fr/ /orion/Telescope/Telesc.html	Mozilla/4.0 (compatible ; MSIE 5.0b1 ; Mac_PowerPC)

Table 1.1 – Fragment d'un fichier log Web contenant 5 requêtes HTTP (unités élémentaires).

L'analyse des fichiers log Web est particulièrement utile car elle fournit des informations sur la manière dont les utilisateurs naviguent réellement sur le site Web. Après la réalisation d'une telle analyse, il est ainsi possible :

- de mettre en évidence les fonctionnalités les plus et les moins utilisées dans le site ;
- de chercher à comprendre les raisons pour lesquelles les fonctionnalités les moins utilisées sont délaissées par les utilisateurs afin, selon les cas, de les améliorer ou de les supprimer ;
- de mettre en évidence les erreurs les plus fréquemment commises par les utilisateurs, afin d'identifier et de résoudre les problèmes d'organisation ou d'utilisation qui pourraient en être la cause.

Le WUM peut encore apporter des avantages à d'autres domaines, comme par exemple, l'ajout dynamique de liens dans des pages Web [99], la recommandation de produits [81, 75], la caractérisation de groupes d'utilisateurs, l'amélioration de politiques comme le *caching* et le *prefetching* anticipés [88], etc.

5.1 Identification de l'utilisateur

En ce qui concerne l'identification de l'utilisateur, pour les sites Web exigeant un login préalable, le nom de l'utilisateur est enregistré dans le troisième champ du fichier log Web. Dans ce cas-ci, cette information peut être utilisée pour identifier l'utilisateur. Autrement, l'identification de l'utilisateur à partir des autres données enregistrées dans le fichier log par le serveur Web devient une tâche assez complexe en raison des serveurs *proxy-cache*, des adresses dynamiques, des utilisateurs multiples (par exemple, dans une bibliothèque, un *cyber* café, etc.) ou encore quand un utilisateur accède à Internet à partir de plusieurs ordinateurs.

Selon [2], les points forts qui justifient l'utilisation d'un serveur *proxy-cache* sont : *(i)* la sécurité, une seule machine accède à Internet ; et *(b)* les transferts, une page demandée par plusieurs personnes est téléchargée une seule fois. Toutefois, en ce qui concerne l'identification de l'utilisateur, la présence d'un serveur *proxy-cache* rajoute des effets secondaires, comme par exemple le fait que les logs ne refléteront plus exactement l'utilisation d'un site Web, une fois que l'adresse enregistrée dans les fichiers log sera celle de la machine *proxy-cache* et non plus celles des machines des clients. De plus, la demande d'une page présente dans le *cache* du serveur *proxy-cache* ne sera plus transmise au serveur qui héberge la page en question. Par conséquent, cette demande ne sera pas enregistrée sur le log du site Web, ce qui rend ce fichier incomplet vis-à-vis des vraies demandes des utilisateurs.

Pour tout cela, l'identification précise des différents utilisateurs connectés au même serveur *proxy-cache* devient impossible. Malgré tout, un certain nombre de techniques ont été proposées afin de contourner les difficultés imposées sur l'identification des utilisateurs. Les techniques les plus communes sont les *cookies* (petits fichiers textes stockés chez le client), le login d'utilisateur et les navigateurs modifiés. Toutes ces techniques ont l'inconvénient de s'introduire dans le domaine privé de l'utilisateur. Cependant, dans [16], les auteurs rapportent que la combinaison des champs *IP Address* et *User Agent* (le navigateur Web) d'un fichier log Web identifie correctement l'utilisateur dans 92.02% des cas et seul un nombre limité de ces combinaisons (1.32%) sont utilisés par plus de trois utilisateurs.

Applicant cette dernière stratégie sur les données du tableau 1.1, il est possible d'identifier un certain utilisateur par le couple adresse IP (194.78.232.8) et navigateur (MSIE 5.0b1) ayant démarré une visite sur le site de l'INRIA à partir de la page d'accueil et en suite demandé trois pages : *liens.htm*, *Telesc.html* et *Vsurv.html*.

5.2 Identification des navigations de l'utilisateur

Une fois l'utilisateur (approximativement) identifié, le problème consiste alors à détecter toutes les requêtes en provenance de cet utilisateur. Concernant cette problématique, la méthode la plus simple pour le groupement des requêtes en sous-ensembles de requêtes appartenant au même utilisateur est d'utiliser le temps de latence maximum entre deux requêtes successives. Les auteurs de [23] ont estimé ce temps de manière empirique à 25.5 minutes. La majorité des méthodes d'analyse de l'usage du Web ont donc adopté le temps de latence de 30 minutes.

L'auteur de [93] définit *navigation* comme l'ensemble de requêtes (clics) en provenance d'un même utilisateur séparées au-delà de 30 minutes, et *session* comme l'ensemble de navigations d'un même utilisateur.

Nous pouvons considérer la combinaison adresse IP plus Navigateur comme étant un critère acceptable pour l'identification d'un utilisateur dans le cadre d'une activité ponctuelle. Cette stratégie serait capable d'identifier les requêtes en provenance d'un même utilisateur dans le contexte d'une seule navigation. Cependant, on ne peut pas généraliser cette combinaison pour l'identification de plusieurs navigations appartenant à un même utilisateur, puisque nous n'avons aucune garantie que l'utilisateur d'avant aura les mêmes valeurs pour le couple adresse IP plus Navigateur lors d'une prochaine visite sur le site Web.

5.3 Prétraitement de données

Le processus du WUM comporte trois étapes principales : le prétraitement des données, la découverte de patterns d'usage et l'analyse des résultats. La phase de prétraitement des données est souvent la plus laborieuse et qui demande le plus de temps, ceci dû en particulier à l'absence de structuration et à la grande quantité de bruit existant dans les données brutes d'usage.

Le prétraitement des fichiers log Web consiste à structurer les données contenues dans ces fichiers afin de les préparer à une future analyse. Les fichiers logs Web étant souvent du type texte, l'un des objectifs de l'étape de prétraitement est de transférer ces données dans un environnement plus facile à exploiter (comme par exemple, dans une base de données). Les objets à identifier dans un processus de prétraitement de fichiers logs Web sont les requêtes des robots Web (souvent déclenchés par les moteurs de recherche), les requêtes issues des utilisateurs humains ainsi que les navigations et les sessions concernées [93].

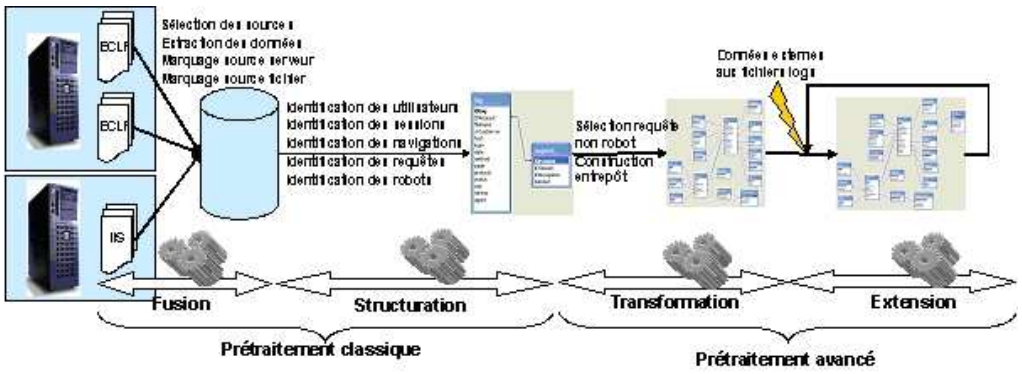

Figure 1.2 – Méthodologie de prétraitement de fichiers logs Web.

Diverses techniques de prétraitement de fichiers log ont été proposées dans la littérature du WUM. Dans cette thèse, nous adoptons la méthodologie proposée par [8] [9] et inspirée des travaux de [32] [94]. Cette approche préconise un prétraitement incrémental temporel et spatial dont le but est de prétraiter les fichiers log dans leur ordre d'arrivée sans avoir à refaire les procédures de prétraitements déjà accomplis [1]. Cette approche prend en compte l'aspect multi-sites, indispensable pour appréhender les pratiques des internautes qui naviguent de façon transparente sur plusieurs sites.

L'approche adoptée applique les étapes de *prétraitement classique* (fusion et de structuration de données) et de *prétraitement avancé* (transformation et extension de données) (cf. figure 1.2, extraite de [8]). Cette approche permet d'extraire des navigations. Une navigation étant définie comme une suite de clics provenant d'un même internaute et séparés au plus de 30 minutes. Elle constitue ainsi la trajectoire d'un internaute sur le site analysé.

Dans l'étape de *fusion* des données, les fichiers logs Web sont sélectionnés et fusionnés, soient-ils issus de différents serveurs Web ou formatés selon différents standards. L'étape de *structuration* concerne l'identification des robots et leurs requêtes ainsi que l'identification des requêtes produites par un être humain, les navigations et les sessions associées. Dans l'étape de *transformation*, les données sont organisées dans un entrepôt sous forme d'un schéma en étoile. Finalement, l'étape d'*extension* des données concerne l'extraction d'information des fichiers log à l'aide des connais-

1. L'outil de prétraitement appliqué a été développé en format *open source* et est disponible pour téléchargement à l'adresse suivante : `https://gforge.inria.fr/projects/webloghousing/`

sances externes (structure et contenu du site, informations additionnelles sur les utilisateurs, etc.).

6 Travaux existants

Dans cette section, nous allons passer en revue les principales approches permettant l'introduction du temps dans la fouille de données Web.

6.1 Approches d'extraction de motifs séquentiels

L'extraction de motifs a été initialement présentée dans [7]. Pour étendre cette problématique à la prise en compte du temps des transactions, les mêmes auteurs ont proposé dans [6] la notion de séquence. Considérons une base de données de séquences où chaque séquence est une liste de transactions triée par le temps et chaque transaction est un ensemble d'items. Le problème consiste à découvrir tous les motifs séquentiels avec un support minimum spécifié par l'utilisateur, où le support d'un motif est défini comme le nombre de séquences qui contient le motif sur toutes les séquences de la base. Un exemple de motif séquentiel serait *"25% de clients achètent les livres A et B dans une transaction, suivi par l'achat d'un livre D dans une transaction postérieure"*.

Quand les transactions analysées proviennent d'une accumulation portant sur une période de temps potentiellement longue (comme dans le cas des fichiers log Web), on s'attend à ce que les motifs évoluent avec le temps. Les motifs séquentiels découverts doivent ainsi être mis à jour continuellement (avec des algorithmes efficaces) afin de suivre fidèlement leur évolution dans le temps. Ceci exige de la méthode une surveillance continue des motifs existants, pré-réquit d'importance essentielle pour les applications centrées sur la dimension temporelle. Une solution possible pour traiter ce problème est définir un schéma approprié pour effectuer le partitionnement du temps. Dans [70], les auteurs appliquent une découpage du flux de données en *batches* de taille fixe. Les auteurs de [71] proposent une méthode de division récursive pour la découverte de motifs séquentiels capables de mettre en évidence l'existence des comportements parfois minoritaires mais pourtant intéressants présents dans les fichiers log Web. Cette technique repose sur des résumés des motifs et les méthodes neuronales.

Dans [26], les auteurs proposent la découverte de motifs surprenant, c'est-à-dire,

de motifs inattendus et donc intéressants dans l'analyse de ventes du marché à partir de l'observation de la variation de corrélation des achats d'articles sur l'échelle du temps. L'inspiration de ce travail est issue de l'analyse de séries temporelles, et l'accent est mis sur la construction d'une subdivision du temps en intervalles de façon à ce que les statistiques sur des règles varient nettement entre deux intervalles consécutifs. Dans [12]n les auteurs proposent un modèle générique de règle (Generic Rule Model, GRM) pour la modélisation du contenu et des statistiques d'une règle en tant qu'objet temporel. Dans des travaux ultérieurs [13] [14], les mêmes auteurs proposent un moniteur de modèles automatisé (PAttern Monitor, PAM) basé sur les mêmes principes de GRM. Ce moniteur considère les patterns en tant qu'objets temporels obéissant à un modèle de changements sur le temps.

Dans [62], les auteurs discutent en quelques lignes des méthodes pour la découverte des modèles séquentiels, des motifs fréquents et des modèles périodiques dans les flux de données. Ils évoquent également des techniques concernant l'analyse statistique de telles approches.

D'autres problèmatiques concernant l'extraction de motifs séquentiels, à savoir le grand nombre de règles découvertes ainsi que l'identification des règles intéressantes, sont des questions largement discutées [30, 67].

6.2 Approches basées sur le raisonnement à partir de cas

Le Raisonnement à Partir de Cas (RàPC) se dit d'une approche de résolution de problèmes basée sur la réutilisation par analogie d'expériences passées appelées *cas*. Un cas est généralement indexé pour permettre de le repérer suivant des caractéristiques pertinentes et discriminantes, appelées *indices*. Les indices déterminent dans quelle situation (ou contexte) un cas peut être de nouveau réutilisé [1, 58].

Le système BROADWAY (**BRO**wsing **AD**visor reusing path**WAY**s) [56] est un assistant pour la navigation sur le Web réutilisant les navigations passées d'un groupe d'utilisateurs. Ce système utilise le RàPC pour proposer des pages à visiter en recherchant des navigations similaires à celles effectuées par l'utilisateur. BROADWAY utilise les navigations des utilisateurs pour en extraire des expériences utiles (cas) permettant d'associer à un comportement (succession de pages visitées), un ensemble de pages qui seront proposées à l'utilisateur. Dans ce système, c'est l'utilisateur qui marque le début et la fin de sa session de recherche au moment de sa demande d'aide. BROADWAY est un serveur HTTP utilisé comme proxy,

il intercepte ainsi toutes les demandes de documents utilisant le protocole HTTP. Durant une session de recherche, BROADWAY peut suggérer un ensemble de documents à l'utilisateur suivant l'état courant de la navigation. Le système permet encore aux utilisateurs d'évaluer ou d'annoter les documents traversés grâce à une barre d'outils insérée dynamiquement dans les pages HTML visualisées. Le système BROADWAY s'appuie, entre autre sur la durée d'affichage des pages Web pour en déduire l'importance. De plus, la solution proposée par ce système consiste en un ensemble de pages triées par ordre de pertinence obtenue par une similitude calculée sur des comportements de navigation passés et ne peut donc être construite que dans le domaine de pages déjà visitées, ce qui limite considérablement les ressources du Web. Le système BROADWAY reste toutefois indépendant du navigateur, ce qui permet son utilisation sur différentes plates-formes.

RADIX (**R**echerche **A**ssistée de **D**ocuments **I**ndexés sur l'e**X**périence) [35] est un système d'aide à la recherche d'information sur le Web et se base sur une description plus détaillée de navigation que le système BROADWAY. L'observation des actions des utilisateurs (telles que la sélection de pages pour le *bookmark*, l'édition de l'adresse des pages, les actions de retour ou avance de pages, etc.) sont utilisées dans la représentation de la navigation de l'utilisateur et ses composants. Pour ce faire, RADIX utilise un navigateur spécifique avec des fonctions adaptées à la surveillance des actions de l'utilisateur. L'utilisation de ce système est par conséquence limitée à une plate-forme spécifique.

Le système CASEP (**CA**se-based reasoning system for **SE**quence **P**rediction) [104] [105] est utilisé pour la prédiction de pages à partir de séquences d'actions d'un utilisateur sur un site Web. Ce système présente quelques limitations, comme par exemple la définition d'un nombre fixe d'états de la séquence courante. Celle-ci et d'autres limitations ont motivé la définition du système hybride CASEP2 [106] par les mêmes auteurs. Ce nouveau système a eu pour but d'apporter des améliorations au précédent système CASEP tout en prenant en compte l'aspect temporel des données ainsi que l'utilisation du système à long terme. Dans CASEP2, il n'y a pas de restriction sur la longueur de la séquence dans la représentation de cas. Dans ce système, l'aspect temporel des données est pris en compte par la modélisation de toute séquence sous la forme d'une matrice de covariance dynamique [103, 102]. Cette matrice prend en compte la distribution des états de la séquence dans l'espace ainsi que leur ordre temporel. La base de cas du système de RàPC est partitionnée en utilisant un réseau de neurones M-SOM-ART ayant les propriétés de stabilité

et de plasticité [102], importantes pour une utilisation à long terme du système. Dans [33,47], les auteurs utilisent les réseaux de neurones dans les différentes phases du cycle RàPC pour traiter des données temporelles.

Le système COBRA (CBR-based **CO**llaborative **BR**owsing **A**dvisor) [69] effectue la prédiction des actions des utilisateurs sur un site Web. COBRA permet de guider un utilisateur à naviguer dans un site et de prédire ses futures pages à visiter tout en réutilisant des traces de navigations passées similaires à la navigation courante. La stratégie utilisée consiste soit de modifier les liens qui relient ses différentes pages, soit de modifier ses contenus sémantiques en fonction de leur utilisation. Pour chaque page demandée, le serveur Web insère un *applet Java* invisible qui envoie des événements au serveur quand la page est visualisée par un client, et ce même si la page est chargée à partir du *cache* client. Cette stratégie permet de calculer le temps de visualisation de page sans prendre en compte le temps de transfert du fichier sur le réseau. Un des avantages de COBRA est la structure de cas et la phase de réutilisation proposées qui rendent possible la prévision d'accès aux pages n'ayant jamais été visitées auparavant.

Le système LETIZIA [66] aide l'utilisateur en déterminant ses centres d'intérêt par l'analyse des pages parcourues et en explorant, pendant le temps de lecture des pages, les pages liées qui semblent le mieux correspondre aux attentes de l'utilisateur. LETIZIA est un agent installé du côté client qui recherche sur le Web des pages semblables à celles que l'utilisateur a déjà visité ou mis dans le *bookmark*. La principale faiblesse de LETIZIA est de ne pas garder trace des sessions de recherche effectuées. Letizia n'apprend pas à partir de ses expériences et établit simplement une description d'intérêt de l'utilisateur pour la session courante en enregistrant ses actions.

Le système HYPERCASE [73] s'appuie sur des recherches prototypiques effectuées par des experts du domaine pour guider la navigation des utilisateurs. Ce système se référant au RàPC ne peut pas apprendre à partir des navigations réelles d'un groupe d'utilisateurs puisque son raisonnement est uniquement basé sur des cas préenregistrés et construits par des experts. Il ne peut pas y avoir d'évolution des propositions dans le temps. L'expérience accumulée par les utilisateurs reste donc inexploitée.

6.3 Approches de classification

Approches pour la détection des changements

Le système DEMON [49] préconise la distinction entre les changements systématiques et non systématiques des données. Le système cherche sur la dimension temporelle les blocs de données qui doivent être traités afin d'extraire de nouveaux patterns. L'algorithme de classification incrémental utilisé est issu de DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [45]. Le système examine quelles parties des clusters actuels sont affectées par une mise à jour de la base de données sur une fenêtre de temps et ajuste les groupes par conséquence. Le but principal est de mettre à jour une base de connaissances par l'intégration des changements repérés.

Le système FOCUS [50] compare deux jeux de données à l'aide d'une mesure de déviation basée sur les modèles induits par les données. Pour la comparaison, les auteurs supposent qu'un modèle est formé par un *composant de structure* et un *composant de mesure*. Par exemple, le composant de structure d'un arbre de décision est une représentation de ses nœuds, alors que le composant de mesure capture la distribution des classes dans chaque nœud. Pour comparer deux patterns, ceux-ci sont décomposés et des régions intéressantes sont identifiées. Ce sont des régions où deux patterns sont en désaccord dans la distribution des classes. Le sous-ensemble de données appartenant à chaque région est alors résumé par le composant de mesure. Les clusters sont considerés comme régions non recouvrantes décrites par un ensemble d'attributs (composant de structure) et correspondant à un ensemble de données (composant de mesure). Dans ce système, l'emphase est sur la comparaison des ensembles de données.

Le système PANDA (PAtterns for Next-generation DAtabase systems) [15] propose des méthodes basées sur la logique d'agrégation pour la comparaison des *patterns simples*, définis sur les données brutes, par exemple un cluster, et des *patterns complexes*, définis sur d'autres modèles, par exemple une partition. La distance entre deux patterns complexes est calculée de manière ascendante basée sur les distances entre les patterns simples qui les composent. Dans le système PANDA, l'emphase est donnée à la comparaison générique et efficace entre n'importe quels patterns complexes.

Le système MONIC (Modeling and Monitoring Cluster) [90] est inspiré des systèmes PANDA et PAM. Dans ce système, une fonction d'âge est appliquée pour

la pondération des données sur le temps. Le système MONIC assume une base de données cumulative sur laquelle une méthode de clustering est réappliqué lors de l'arrivé des nouvelles données, ce qui demande beaucoup de ressources physiques quand la base de données assume des dimensions importantes. Dans ce système, aucune stratégie de réduction ou résumé des données n'est appliquée.

Approches basées sur le clustering spatio-temporel

Les auteurs de [77] ont étudié la stabilité des clusters à partir de l'observation des régions spatiales. Dans leur approche, aucune méthode de clustering n'est appliquée, un cluster correspond alors à la négation de l'hypothèse d'homogénéité. La procédure de détection de clusters consiste à identifier des régions spatiales qui présentent, pour une certaine propriété, de valeurs plus grandes que prévues. L'ensemble de régions est statique et connu à l'avance. Les valeurs prévues sont calculés à partir de l'analyse des séries chronologiques des valeurs passées. Une fenêtre de temps glissante est adoptée pour la détection des clusters émergents, persistants et évolutifs.

Les auteurs de [100] ont proposé une méthode pour la détection des changements sur des clusters générés à partir de données scientifiques. Ils ont étudié les patterns d'association d'objets spatiaux (SOAP, Spatial Object Association Patterns). Un SOAP est caractérisé par le nombre de 'aperçus instantanés' des données, où il se produit et le nombre d'exemples dans un aperçu instantané. Avec cette information, l'algorithme détecte des événements de *formation*, de *dissipation* et de *continuation* des clusters. Leur méthode n'est pas forcement consacrée à l'analyse des clusters, elle se rapporte plutôt à des patterns en général.

Pour [4], un cluster correspond à la densification des données dans un espace multidimensionnel, où chaque dimension correspond à un attribut. Des exemples intuitifs pour ce type de définition de cluster sont facilement trouvés dans les Systèmes d'Information Géographique (SIG). Par exemple, une ville peut être vue comme une zone où la concentration de maisons est plus dense que dans son entourage. L'évolution de la ville (par exemple, sa croissance ou rétrécissement) peut être modélisée en tant que changement de densité. L'auteur rapproche des clusters avec des fonctions à noyau et calcule des changements de densité du noyau à chaque point spatial. Des changements de densité aux points spatiaux voisins sont agrégés, de sorte que l'on puisse détecter les secteurs qui changent à vitesse différente de leurs voisinages.

Pour chaque point au temps t, la densité future correspond au changement de densité du noyau après l'instant t, alors que la densité passée correspond au changement de densité du noyau avant l'instant t. La différence entre ces deux densités calculées constitue la vitesse des changements [4]. L'auteur distingue différents types de changements et l'accent est mis sur *(i)* la vitesse du changement et *(ii)* les points présentant la vitesse la plus élevée - les épicentres. Un dispositif particulier de sa méthode est l'identification des propriétés des données qui contribuent le plus aux changements.

Toutes les approches de cette catégorie opèrent sur une trajectoire et supposent que celle-ci ne change pas. Par conséquent, ces méthodes ne peuvent pas être combinées à tous les types d'algorithmes de groupement, par exemple les algorithmes hiérarchiques ou les algorithmes basés sur la densité. Les algorithmes de groupement hiérarchique emploient une ultramétrique, ainsi leurs clusters ne peuvent pas être étudiés en dehors de l'espace métrique qui les a produits. De plus, ces méthodes juxtaposent chaque cluster à la trajectoire et ne peuvent pas tracer des interférences parmi les clusters, par exemple l'absorption (fusion) d'un cluster par un autre.

Approches pour la détection de changements sur un flux de texte

Les auteurs de [5] proposent le concept de *droplet* comme le résumé d'un cluster issu d'un flux de texte. Un *droplet* se compose de deux vecteurs, un vecteur contenant tous les mots apparaissant dans le cluster fondamental et un autre contenant toutes les paires de mots en co-occurrence. Les auteurs utilisent le concept de *droplet* pour suivre l'évolution des clusters. Un nouveau document est affecté au cluster dont le droplet est le plus similaire. Si un cluster ne reçoit pas des documents pendant un certain temps, il devient inactif. Si un document ne peut pas être assigné à aucun cluster, ce document devient lui-même un cluster, remplaçant de ce fait le cluster inactif le plus ancien. Cette approche suppose que le nombre de droplets (thèmes dans un flux) soit fixe et que les plus anciens soient remplacés par les plus jeunes.

Les tendances des clusters sur un flux de texte sont également adressées par [72]. L'emphase repose sur l'analyse des thèmes et leur évolution. Les auteurs appliquent une stratégie de groupement probabiliste, dans laquelle chaque document contribue à chaque cluster avec une probabilité qui augmente avec sa similarité à la moyenne du cluster. Un thème correspond à la moyenne d'un cluster et comprend les mots

qui caractérisent le plus les individus du cluster. Pour le suivi des tendances, les auteurs relient les clusters présentant des thèmes similaires. Ceci résulte à un graphe d'évolution de thème, où une séquence de clusters constitue un thème persistance.

Approches de classification basées sur les modèles

En outre, nous avons les approches de classification basées sur des modèles. Ces approches ont été utilisées dans beaucoup d'applications concernant les données du Web [11]. Dans ces modèles, le nombre de groupes est déterminé à travers des méthodes probabilistes, telles que BIC (critère bayésien de l'information), approximations bayésiennes, ou méthodes de *bootstrap* [48]. La structure du modèle peut être déterminée par les techniques de sélection de modèle et l'estimation de paramètres en utilisant des algorithmes de vraisemblance maximale, par exemple l'algorithme Expectation-Maximization [38]. Les modèles de Markov (cachés ou du premier ordre) [11, 21] sont les modèles les plus indicatifs dans ce contexte.

6.4 Approches basées sur l'Analyse de Données Symboliques (ADS)

L'Analyse de Données Symboliques (ADS) [18] [43] généralise l'analyse de données classiques (AD) à de nouveaux types de données en établissant un cadre de modélisation mathématique à base de types de données structurées allant au-delà de l'expressivité tabulaire classique. Dans l'analyse de données symboliques, les données dites *symboliques* conservent la description de la réalité dans toutes ses variations. Ainsi, dans un tableau de données symboliques, il n'y a plus, comme en statistique classique, une valeur par case, mais une statistique par case (intervalle, histogramme, distribution, etc.). L'ADS a pour objectif, dans un premier temps, de constituer des concepts à partir de la table initiale des individus décrits par des variables classiques. Ces concepts sont décrits par des variables symboliques capables d'exprimer la variation interne des données par des intervalles, variables à valeurs multiples, histogramme, distribution de probabilité, etc. Un concept est défini par une *intension* et une *extension*. L'intension est un ensemble de propriétés caractéristiques du concept, l'extension est l'ensemble des individus (instances) du concept qui satisfont ces propriétés. L'une des avantages de l'ADS concerne sa capacité de réduction de la taille des données en considérant des unités d'étude d'un plus haut niveau d'abstraction (ex. classes) plutôt que des individus isolés.

Par rapport aux approches classiques, l'analyse des données symboliques présente les caractéristiques suivantes :

- Elle s'applique à des données plus complexes. En entrée elle part de données symboliques (variables à valeurs multiples, intervalle, histogramme, distribution de probabilité, de possibilité, capacité etc.) munies de règles et de taxonomies et peut fournir en sortie des connaissances nouvelles sous forme d'objets symboliques.
- Elle utilise des outils adaptés à la manipulation d'objets symboliques de généralisation et spécialisation, d'ordre et de treillis, de calcul d'extension, d'intension et de mesures de ressemblances ou d'adéquation tenant compte des connaissances sous-jacentes basées sur les règles et taxonomies.
- Elle fournit des représentations graphiques exprimant entre autres la variation interne des descriptions symboliques. Par exemple, en analyse factorielle, un objet symbolique sera représenté par une zone (elle même exprimable sous forme d'objet symbolique) et pas seulement par un point.

Les principaux avantages des objets symboliques peuvent se résumer comme suit :

- Ils fournissent un résumé de la base plus riche que les données agrégées habituelles (car tenant compte de la variation interne et des règles sous-jacentes aux classes décrites, ainsi que des taxonomies fournies, on est loin des simples centres de gravités.)
- Ils sont explicatifs, puisqu'ils s'expriment sous forme de propriétés des variables initiales ou de variables significatives obtenues (axes factoriels), donc en termes proches de l'utilisateur.
- En utilisant leur partie descriptive, ils permettent de construire un nouveau tableau de données de plus haut niveau sur lequel une analyse de données symbolique de second niveau peut s'appliquer.
- Afin de modéliser des concepts, ils peuvent aisément exprimer des propriétés joignant des variables provenant de plusieurs tableaux associés à différentes populations. Par exemple, pour construire un objet symbolique associé à une ville, on peut utiliser des propriétés issues d'une relation décrivant les habitants de chaque ville et une autre relation décrivant les foyers de chaque ville.
- Plutôt que de fusionner plusieurs bases pour étudier ensuite la base synthétique obtenue, il peut être plus avantageux d'extraire d'abord des objets sym-

boliques de chaque base puis d'étudier l'ensemble des objets symboliques ainsi obtenus.
- Ils peuvent être facilement transformés sous forme de requête d'une base de données.
- Ils peuvent donc propager les concepts qu'ils représentent d'une base à une autre (par exemple, d'un pays à l'autre de la communauté européenne).

Un certain nombre de méthodes ont été adaptées/développées pour traiter les données symboliques. Le logiciel SODAS (Symbolic Official Data Analysis System) [96] [2] est un puissant outil réunissant diverses méthodes d'extraction, analyse et visualisation des données symboliques. Entre autres, nous pouvons citer la méthode STAT qui calcule les statistiques de base sur de données symboliques [17]. La méthode PCM propose une extension de l'analyse en composantes principales à des données du type intervalle [24] [31]. Cette méthode produit la représentation graphique et les facteurs tout en prenant compte de la variation des intervalles et de leurs valeurs centrales. La méthode HIPYR [20] effectue une classification hiérarchique ou pyramidale, sur un ensemble de données symboliques, basée sur le tableau de données ou sur une matrice de dissimilarités. La méthode SCLUST [65] représente une extension de la méthode des Nuées Dynamiques [40] aux données symboliques. La méthode DIV [29] [27] de classification descendante hiérarchique produit des classes ainsi que leur description symbolique à partir de divisions successives basées sur la variable symbolique produisant la meilleure séparation de l'ensemble de données. Dans cette méthode, la meilleure division est celle qui produit les deux classes minimisant la somme d'une mesure de dissimilarité symbolique entre les individus de chaque classe.

Pour la visualisation des données symboliques, quelques solutions ont été proposées pour représenter graphiquement les objets symboliques. Zoom Star [78] [79] permet de représenter un objet symbolique en 2D ou 3D. Temporal Star [80] permet de représenter un objet symbolique à différentes époques. Des étoiles en perspectives sont enfilées sur un axe représentant le temps. Chaque axe d'une étoile représente une variable. Le graphique peut être déplacé, tourné, zoomé. Il est également possible de changer la couleur des axes, de choisir les variables et les périodes à représenter.

2. Ce logiciel peut être téléchargé à l'adresse suivante : `http://www.ceremade.dauphine.fr/~touati/sodas-pagegarde.htm`

6.5 D'autres approches

La problématique de l'insertion du temps dans l'analyse des données Web a également été explorée sous l'optique d'autres techniques telles que : logique floue [107,92], abstractions temporelles [76], arbres de décision [55], arbres de concepts [3] et graphes [39].

Dans [107], les auteurs proposent une approche basée sur un modèle de treillis d'usage du Web (Web Usage Lattice Model) qui utilise une hiérarchie d'activités d'accès du Web décrites par le moyen de la logique floue pour représenter des concepts temporels tels que le matin, l'après-midi, le soir et les catégories significatives de pages Web. Dans [92], les auteurs présentent un schéma de maintenance du profil de l'utilisateur Web basé sur un algorithme de clusterisation flou (Relational Fuzzy Subtractive Clustering algorithm, RFSC) capable d'ajouter de nouvelles données d'usage à un modèle préexistant sans le coût d'une remodélisation complète. Les auteurs définissent cependant une mesure quantitative qui indique quand la remodélisation complète doit être exécutée afin d'éviter une dégradation du modèle. Leurs résultats montrent que la technique adoptée est presque aussi bonne que celle utilisant la remodélisation complète.

Dans [76], les auteurs proposent l'utilisation des abstractions temporelles définies sur des intervalles de temps. Puis, ils définissent une ontologie pour représenter la connaissance temporelle extraite à partir des données. Ils proposent d'exploiter les résultats obtenus de l'abstraction temporelle dans l'étape de prétraitement des techniques de la fouille de données afin de développer un genre d'analyse intelligente du domaine médical. La problématique de cette approche est celle de la fouille d'ordre supérieur (higher order mining). Les auteurs défendent que l'abstraction temporelle permet une réduction potentielle sur la quantité de données et de bruit.

Dans [55], les auteurs présentent un algorithme basé sur le VFDT (Very Fast Decision Tree Learner) pour la construction incrémentale d'un arbre de décision pour la fouille sur les flux de données à travers un apprentissage défini sur des fenêtres glissantes. Les auteurs de [3] ajoutent à l'analyse statistique classique des logs une information de *concept* rattachée à chaque page. L'objectif est de prendre en compte un changement de centre d'intérêt de l'utilisateur au cours de la session, de pré-segmenter les sessions sur la base des contenus visités et de mieux prédire les liens qui seront suivis en fonction de la position dans un arbre de concepts. Les auteurs attestent une augmentation significative du taux de prédiction en ayant

recours à cette méthode.

Les auteurs de [39] construisent des graphes d'utilisation d'un site Web à partir de l'analyse des fichiers log du centre d'informatique de l'université de Minnesota afin d'étudier leur évolution sur le temps à travers de la fouille des sous-graphes séquentiels.

7 Problèmes ouverts

Les enjeux qui tentent de relever la fouille des données temporelles est de pouvoir prendre en compte simultanément des informations disponibles concernant les entités qui constituent les données de nature temporelle et les données de nature non temporelle. Cela suppose d'être capable d'intégrer des informations de nature différente et de pouvoir ainsi les rattacher à une même catégorie sémantique. Sur le plan méthodologique, il s'agit donc de définir une mesure de ressemblance ou de dissemblance entre deux objets dont la description est fournie par un ensemble complexe d'informations. Jusqu'à présent, les approches proposées ont surtout été fondées sur la juxtaposition de similarités ou de dissimilarités partielles. Il serait donc intéressant d'avoir une mesure de distance, à la fois capable de prendre en compte l'aspect temporel et de traiter les données classiques (non temporelles).

L'une des difficultés les plus importantes liées à la fouille d'usage du Web est la pénurie de *benchmarks* de données d'usage pour l'application et comparaison de différentes techniques. Cela est dû au fait que les données d'usage contiennent des informations privées souvent maintenues sous la propriété des entreprises.

De nos jours, les sites Web deviennent de plus en plus grands parce que plus de pages Web sont ajoutées que supprimées. En conséquence, nous avons une croissance explosive de la taille des fichiers log Web. Le traitement de fichiers logs requiert donc des efforts supplémentaires. Les propriétaires de sites Web optent pour garder les vieilles pages Web parce que le stockage d'information est relativement peu coûteux de nos jours et aussi parce qu'ils savent que leurs clients marquent les pages Web dans leurs *bookmarks* et comptent toujours les trouver en ligne, ce qui est tout à fait compréhensible dans le cadre de la concurrence commerciale. Cependant, il existe des stratégies qui peuvent résoudre ce problème, par exemple, il y a des moyens de faire un changement automatique de pages lorsqu'un client demande une adresse obsolète. Ceci pourrait d'une certaine façon éviter les informations redondantes et

minimiser le volume des fichiers logs enregistrés par le serveur Web.

En ce qui concerne les motifs fréquents, leur extraction se conçoit sur un ensemble fini de résultats possibles (l'ensemble de combinaisons entre les items enregistrés). Ceci n'est pas le cas pour les motifs séquentiels où l'ensemble de résultats est infini. En fait, en raison de l'aspect temporel des motifs séquentiels, un item peut être répété sans limites menant à un nombre très grand de séquences potentiellement fréquentes. Dans ce contexte, la fouille d'ordre supérieur (Higher Order Mining) dans laquelle la fouille est appliquée aux règles précédemment extraites, est un secteur prometteur car il permet de réduire les coûts généraux de la fouille de données [83].

La prise en compte de la dimension temporelle s'avère également nécessaire dans le domaine de la recommandation de produits (pages, livres, films, etc.). En effet, le goût d'un utilisateur peut varier avec le temps. Supposons, par exemple, un groupe d'utilisateurs qui possèdent une préférence pour les films d'horreur en général mais qui, pendant l'hiver, optent pour des comédies. Le système doit être suffisamment habile pour détecter le changement d'intérêt d'un utilisateur et ajuster les recommandations aux besoins de ses utilisateurs. Cependant, d'autres problèmes sont engendrés par la prise en compte des changements de préférences des utilisateurs des systèmes de recommandation. La vulnérabilité d'un système de recommandation constitue une des issues débutantes dans ce secteur. Quelques commerçants sans scrupules profitent de cette faiblesse pour "tromper" le système afin d'avoir leurs produits recommandés plus souvent que ceux de leurs concurrents. Cette problématique est traitée sous la nomenclature de : *shilling attacks* [61].

8 Synthèse

Dans ce chapitre, nous avons abordé quelques points touchant la problématique du traitement de la dimension temporelle dans l'analyse des données. Les questions abordées montrent la nécessité de définition de nouvelles méthodes ou d'adaptation de méthodes existantes pour l'extraction des connaissances et le suivi le changement des données évolutives. Bien qu'il existe de nombreuses méthodes performantes d'extraction de connaissances, peu de travaux ont été consacrés à la problématique de données temporaires et évolutives.

L'état de l'art présenté dans ce chapitre a pu être à l'origine d'un article du type survol (voir publication numéro 1 dans l'annexe B).

Chapitre 2

Classification non supervisée

1 Introduction

La classification non supervisée (classification automatique, regroupement ou *clustering*, en anglais) a pour but de regrouper des individus en classes homogènes en fonction de l'analyse des caractéristiques qui décrivent les individus. Par classes homogènes, on entend regrouper les individus qui se ressemblent et séparer ceux qui sont dissemblables. Dans cette thèse, nous nous intéressons particulièrement à deux grandes familles de méthodes de classification non supervisée, à savoir : les méthodes hiérarchiques et les méthodes de partitionnement, décrites plus en détails dans les sections 2 et 3 de ce chapitre.

L'expression *non supervisée* fait référence au fait qu'aucun superviseur ou label est utilisé pour préciser à quelle classe appartient un individu. En conséquence, le nombre de classes existant dans un ensemble d'individus est a priori inconnu. De ce fait, l'un des problèmes les plus délicats à propos des méthodes de classification non supervisée concerne le choix du nombre de classes à retenir. Pour palier cet écueil, il existe des artifices permettant d'approcher le bon nombre de classes, ceux-ci seront discutés dans la section 4. Une fois le nombre de classes choisi, le prochain pas dans le processus de classification consiste à évaluer la qualité de la partition obtenue par la méthode de classification. Dans les travaux de la présente thèse, nous nous concentrons sur les indices de comparaison de partition basés sur l'extension, présentés dans la section 5. Lorsqu'une partition a été retenue, il convient de caractériser les classes constituant cette partition. Nous nous intéressons notamment à l'interprétation statistique des classes en fonction des variables les plus discriminantes.

Les indices permettant d'identifier le pouvoir discriminant des variables vis-à-vis de chaque classe sont présentés dans la section 6.

2 Méthodes hiérarchiques

La classification hiérarchique est une famille de techniques qui génèrent des suites de clusters emboîtés les uns dans les autres. Les techniques de classification existantes dans cette famille peuvent être classées dans deux grands groupes : *Classification Ascendante Hiérarchique* (CAH) ou *par agrégation* et *Classification Descendante Hiérarchique* (CDH) ou *par division.*

Dans le groupe CAH, tous les individus sont initialement considérées comme des clusters ne contenant qu'un seul individu (singleton). A chaque étape du processus de classification, les deux clusters les plus proches au sens d'une mesure d'agrégation sont fusionnés. Le processus s'arrête quand les deux clusters restants fusionnent dans l'unique cluster contenant tous les individus.

Les techniques dans le groupe CDH partent de la partition triviale à un seul cluster (contenant toutes les observations). Le processus de classification suit en effectuant de divisions successives des clusters jusqu'à ce que soit atteint la partition triviale où chaque observation est un cluster. La division d'un cluster s'opère de façon à ce que la mesure d'agrégation entre les deux clusters descendants soit la plus grande possible, de manière à créer deux clusters bien séparés.

Les clusters résultants des méthodes de classification hiérarchiques sont souvent représentés par une structure graphique appelé *dendrogramme.* Les typologies qui ont le plus de chance d'être significatives sont obtenues simplement en traçant une ligne horizontale en travers du dendrogramme, et en retenant dans la typologie les clusters terminaux juste au-dessus de cette ligne. En changeant la hauteur de la ligne, l'on change le nombre de clusters retenus. Cette pratique constitue un moyen simple pour faire varier la granularité de la typologie finale.

2.1 Classification Ascendante Hiérarchique (CAH)

Dans ce groupe, il existe plusieurs méthodes d'agrégation possibles entre deux clusters. Le choix de la méthode dépend de la problématique que l'on se pose. En ce qui concerne la distance entre deux individus, la distance euclidienne est la plus utilisée en général. Parmi les méthodes d'agrégation existantes pour chaque couple

de clusters, nous citons ci-après les plus classiques :

- **agrégation selon le lien minimum (single linkage)** : l'agrégation entre deux clusters A et B correspond au minimum des distances entre un élément du cluster A et un élément du cluster B, $d(A,B) = \min_{i \in A, j \in B} d(x_i, x_j)$
- **agrégation selon le lien maximum (complete linkage)** : l'agrégation entre deux clusters A et B correspond au maximum des distances entre un élément du cluster A et un élément du cluster B, $d(A,B) = \max_{i \in A, j \in B} d(x_i, x_j)$
- **agrégation selon le lien moyen (average linkage)** : l'agrégation entre deux clusters A et B correspond à la moyenne des distances entre un élément du cluster A et un élément du cluster B, $d(A,B) = \frac{1}{|A||B|} \sum_{i \in A, j \in B} d(x_i, x_j)$
- **la méthode de Ward** : on choisira de regrouper les deux clusters qui génèrent le gain d'inertie intra-classe minimum.

Selon le théorème de Huygens, l'inertie total T d'un ensemble d'individus E est égale à la somme des inerties intra-classe W et inter-classe B de cet ensemble d'individus regroupés dans une partition P (cf. équation 2.1).

$$T(E) = W(P) + B(P) \tag{2.1}$$

où

$$\begin{aligned}
T(E) &= \sum_{i=1}^{n} w_i d^2(x_i, g) \\
W(P) &= \sum_{l=1}^{K} \sum_{i \in C_l} w_i d^2(x_i, g_l) \\
B(P) &= \sum_{l=1}^{K} \mu_l d^2(g_l, g) \\
g &= \sum_{i=1}^{n} w_i x_i \\
g_l &= \frac{1}{\mu_l} \sum_{i \in C_l} w_i x_i
\end{aligned}$$

L'inertie intra-classe d'une partition mesure l'homogénéité de l'ensemble de classes. Une classe sera d'autant plus homogène que son inertie intra-classe sera faible. L'inertie inter-classe d'une partition mesure la séparation entre les classes qui la composent. Plus l'inertie inter-classe d'une partition est grande plus les classe

sont distinctement séparées. Deux cas particuliers sont à considérer :

- Si chaque classe de la partition contient un seul individu, c'est-à-dire, $K = n$, alors $W(P) = 0$ et $B(P) = T(E)$.
- Si la partition n'est composée que d'une seule classe, c'est-à-dire, $K = 1$, alors $B(P) = 0$ et $W(P) = T(E)$.

L'inertie intra-classe d'une partition croit avec la diminution du nombre de classes, où encore, l'inertie intra-classe décroit avec l'augmentation du nombre de classes dans la partition.

Le but de l'algorithme CAH est de passer de la meilleure partition de k classes à la meilleure partition de $(k-1)$ classes selon le critère d'agrégation adopté. Dans cette thèse, nous avons adopté la liaison de Ward comme critère d'agrégation entre deux classes. Selon ce critère, pour passer d'une partition P^K à une partition P^{K-1}, on choisira de regrouper les deux clusters qui génèrent le gain d'inertie intra-classe minimum. Ce gain est calculé de la façon suivante :

$$Gain = W(P^{k-1}) - W(P^k) \tag{2.2}$$

Soit A et B deux classes, μ_A et μ_B leurs respectifs poids. Le gain d'inertie intra-classe engendré par le regroupement de ces deux classes est donné par l'écart de Ward (cf. équation 2.3). L'écart de Ward peut être vu comme une distance entre deux classes si la distance entre deux individus est la distance euclidienne.

$$Gain(A, B) = \frac{\mu_A \mu_B}{\mu_A + \mu_B} d(g_A, g_B)^2 \tag{2.3}$$

où

$$\mu_A = \sum_{i \in A} w_i \quad et \quad g_A = \frac{1}{\mu_A} \sum_{i \in A} w_i x_i$$
$$\mu_B = \sum_{i \in B} w_i \quad et \quad g_B = \frac{1}{\mu_B} \sum_{i \in B} w_i x_i$$

3 Méthodes de partitionnement

Les méthodes de partitionnement ont pour but de fournir une partition des éléments à classer. Le nombre de classes dans la partition à générer doit être fixé au départ. Le principe des méthodes de partitionnement est le suivant : à partir d'un

ensemble E de n individus, on cherche à construire une partition $P = (C_1, ..., C_K)$ contenant K classes homogènes et distinctes, tout en respectant un critère basé sur une distance entre les individus. Doivent se trouver dans une même classe les individus qui se ressemblent et en classes différentes les individus divergents en termes du critère adopté.

Une partition optimale peut être obtenue à partir de l'énumération exhaustive de toutes les partitions, ce qui devient cependant prohibitif en termes de temps de calcul. Comme solution alternative à ce problème, des méthodes de partitionnement basées sur l'optimisation itérative du critère à respecter permettent d'obtenir des groupes bien distincts en un temps de calcul raisonnable. Ces méthodes d'optimisation utilisent une réaffectation afin de redistribuer itérativement les individus dans les K classes.

3.1 La Méthode des Nuées Dynamiques (MND)

Soit $E = \{1, \ldots, n\}$ un ensemble de données de taille n et $\mathcal{P}_K(E)$ l'ensemble de partitions de E en K classes non vides. La Méthode des Nuées Dynamiques (MND de [40] [42]) est basée sur trois fonctions :

- $\boldsymbol{g}$ est une fonction de représentation d'une partition en K classes non vides de E. Dans notre cas, la représentation est le vecteur des K moyennes.
- $\boldsymbol{f}$ est une fonction qui permet d'affecter un prototype L à une partition de E.
- $\boldsymbol{W}$ est une fonction qui permet de mesurer l'adéquation d'une partition P et un prototype L.

L'objectif de la MND est d'optimiser le critère W. Le problème d'optimisation se pose alors en termes de recherche simultanée de la partition $P = (C_1, ..., C_K)$, dans l'espace des partitions $\mathcal{P}_K(E)$, et de la représentation de cette partition $L = (L_1, ..., L_i, ..., L_K)$ dans l'espace de représentations des partitions $\mathcal{L}_K$, ayant la meilleur représentation possible au sens du critère W. En fait, le critère W est une application de $\mathcal{L}_K \times \mathcal{P}_K(E)$ dans $\Re^+$. Dans notre cas, l'espace des prototypes $\mathcal{L}_K \equiv (\Re^p)^K$ et $L_i \in \Re^p$, L_i étant le centre de gravité (moyenne) de la classe i. Le critère W est l'inertie intra-classe qui est positive et dont une petite valeur exprime une bonne adéquation entre L et P.

Pour résoudre le problème d'optimisation du critère W, la MND utilise de façon itérative, la fonction de représentation $g : \mathcal{P}_K(E) \rightarrow \mathcal{L}_K$, qui permet d'associer à une partition de $\mathcal{P}_K(E)$ un élément $\mathcal{L}_K$, et la fonction d'affectation $f : \mathcal{L}_K \rightarrow \mathcal{P}_K(E)$

qui permet d'associer à un élément de $\mathcal{L}_K$ une partition de $\mathcal{P}_K(E)$ (cf. figure 2.1).

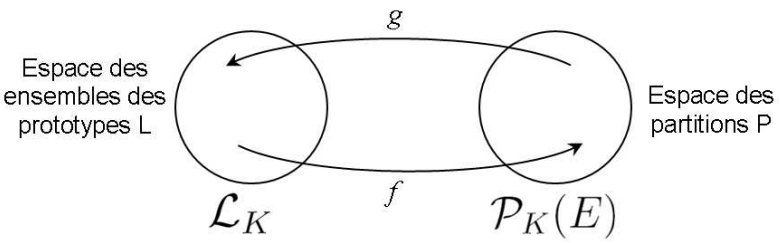

Figure 2.1 – Schéma des fonctions f et g de la MND.

La MND est, en effet, une généralisation d'un ensemble de procédés appelés *"Itération relocation procédure"* [34] ou k-means [68]. Ce procédé s'annonce comme suit. On part de K points de $\Re^2$ ou de K points de E. Ces K points jouent le rôle de prototypes ou centres des classes. Tous les points de E sont affectés au centre le plus proche suivant l'une des techniques suivantes :

a) les K centres sont recalculés quand tous les points de E sont classés.

b) les K centres sont recalculés après l'affectation de chaque élément de E. Un point est ajouté à la classe et sa moyenne est ajustée de manière à prendre en compte le nouveau point.

La MND consiste à entrer E en mémoire centrale et à utiliser successivement et de manière alternative les fonctions f et g jusqu'au moment où une position d'équilibre est atteinte.

E. Diday a également proposé dans [41] une autre méthode appelée Méthode des Nuées Dynamiques Séquentialisée (MNDS). La MNDS est basée sur le même principe que la MND mais ne nécessite pas la mise en mémoire centrale du tableau de données associé à E. La MNDS consiste à actualiser les centres des classes dès que tous les individus d'un paquet sont affectés, ce qui implique un gain de temps important engendré par la réduction du nombre de centrages. E. Diday démontre dans le *Théorème 2* (page 36 de [41]) que les algorithmes MND et MNDS font décroître le même critère W et convergent vers les mêmes solutions.

L'exécution de l'algorithme de la MND aussi bien que de celui de la MNDS se termine lorsqu'un nombre maximum d'itérations est atteint où lorsque le partitionnement ne change plus, c'est-à-dire, lorsqu'aucun individu ne change de classe. La configuration finale obtenue par ces algorithmes est dépendante des prototypes initiaux et peut être arrêté par des minima locaux. Il faut donc exécuter plusieurs fois

l'algorithme et comparer les résultats de manière à extraire la meilleure configuration selon le critère W. Dans [68], ce critère est défini comme l'erreur k-means et correspond à la somme des distances au carré entre chaque individu et le prototype de la classe à laquelle il est affecté (cf. équation 2.4).

$$E_{k-means} = \sum_{l=1}^{K} \sum_{i \in C_l} w_i d^2(x_i, g_l) \tag{2.4}$$

3.2 Les cartes auto-organisatrices de Kohonen

Les cartes auto-organisatrices de Kohonen (ou Self Organizing Maps (SOM), en anglais) [57] sont généralement présentées comme une méthode de classification qui généralise les méthodes du type partitionnement en introduisant la notion de voisinage entre les classes. Les classes sont représentées par des neurones disposés sur un réseau, appelé *carte*. Cette carte définit un voisinage a priori entre les classes en respectant une topologie préétablie. L'apprentissage de la méthode aboutit à la concordance entre la proximité des neurones sur la carte et la proximité des individus dans l'espace des données : deux individus associés à des neurones voisins sont proches dans l'espace des données.

Un réseau de Kohonen est constitué de : *(i)* une couche d'entrée, et *(ii)* une couche de sortie (ou couche de compétition). Tout individu à classer est représenté par un vecteur multidimensionnel présenté à la couche d'entrée. Les neurones de la couche de sortie entrent en compétition à chaque présentation d'individu, cependant un seul neurone est élu neurone vainqueur. Les neurones voisins du neurone vainqueur ont une action d'excitation, alors que les neurones en dehors de ce voisinage ont une action d'inhibition. Le principe de l'apprentissage compétitif consiste à récompenser le vainqueur, c'est-à-dire, à rendre celui-ci encore plus sensible à une présentation ultérieure du même individu. Dans le processus d'éxecution des cartes de Kohonen, une *présentation* correspond à l'action de présenter un individu du jeu de données au réseau de Kohonen, alors qu'une *itération* correspond à la présentation de tous les individus du jeu de données au réseau de Kohonen.

L'algorithme des cartes auto-organisatrices de Kohonen se décline de la façon suivante :

– **Initialisation** :

  Prétraitement des données : chaque vecteur d'entrée est centré et réduit de telle façon à ce que son écart type soit égal à 1.

Choix des prototypes : initialisation aléatoire des prototypes initiaux $m_i^0, i = 1, ..., m$. Les poids initiaux des neurones de compétition sont également normalisés à 1.

- **Présentation des données** : il est préférable que les individus soient présentés de façon aléatoire avec l'alternance d'individus de différentes classes.
- **Recherche du neurone vainqueur** : l'algorithme recherche le neurone dit vainqueur définit comme suit : $\|x_i - m_c\| = \min_l \|x_i - m_l\|$
- **Mise à jour des prototypes des classes** : Après chaque présentation d'individu, tous les prototypes des classes sont modifiés selon l'équation 2.5, de façon à les rapprocher de l'individu présenté au réseau. La pondération qui détermine l'ampleur des modifications de position dans l'espace est fonction de la distance sur la carte entre le neurone vainqueur et le neurone considéré.

$$m_l^{t+1} = m_l^t + \alpha(t)h(c, l, t)w_i(x_i - m_l^t) \tag{2.5}$$

Dans l'équation 2.5, α est le taux d'apprentissage et doit décroître au cours du temps. Il est souvent défini par une fonction inversement proportionelle au temps, comme montre l'équation 2.6. Dans cette équation, t est l'itération actuelle, α_0 est le pas d'apprentissage initial et C est une constante arbitraire.

$$\alpha(t) = \alpha_0 \frac{C}{C + t} \tag{2.6}$$

Dans l'équation 2.5, $h(c, l, t)$ est appelée fonction de voisinage. Cette fonction est souvent gaussienne (cf. équation 2.7) et correspond à la pondération du neurone l lorsque le vainqueur est le neurone c.

$$h(c, l, t) = e^{-\frac{d(r_c, r_l)^2}{2\sigma(t)^2}} \tag{2.7}$$

Dans l'équation 2.7, $d(r_c, r_l)$ correspond à la distance entre les neurones c et l sur la carte, $\sigma(t)$ correspond au rayon de voisinage et est défini selon l'équation 2.8. Dans cette dernière équation, σ_0 est le rayon de voisinage initial, T est le nombre total d'itérations et t est l'indice de l'itération actuelle. Le rayon de voisinage décroit au cours du temps.

$$\sigma(t) = \sigma_0 \frac{T - t}{T} \tag{2.8}$$

– **Test de convergence** : Il est judicieux de continuer l'exécution jusqu'à ce que le nombre d'itération soit atteint, puisque l'algorithme peut continuer à converger même en absence de changements d'affectation. L'erreur de distorsion moyenne (ou critère de Kohonen) correspond au critère optimisé par l'algorithme de gradient stochastique. Ce critère est défini selon équation 2.9 et doit être calculé après chaque itération (présentation du jeu de données). Lorsque le voisinage se réduit au seul neurone vainqueur, le critère de Kohonen tendra vers l'erreur des k-means.

$$E_{Kohonen} = \sum_{i=1}^{n} \sum_{l=1}^{m} h(c,l) w_i d^2(x_i, m_l) \tag{2.9}$$

Il est également possible d'adapter l'erreur K-means (équation 2.4) aux cartes auto-organisatrices de Kohonen comme montre l'équation 2.10, où m_{ic} correspond au prototype de la classe à laquelle l'individu i est affectée lors de sa présentation au réseau. Dans les cartes auto-organisatrices de Kohonen, le prototype de la classe ne correspond pas nécessairement à son centre de gravité.

$$E_{k-means} = \sum_{i=1}^{n} w_i d^2(x_i, m_{ic}) \tag{2.10}$$

Initialisation par l'Analyse en Composants Principales (ACP)

Cette méthode d'initialisation de la carte de Kohonen fait appel à la technique d'Analyse en Composants Principales (ACP) [87, 63]. En effet, le plan d'inertie principal fournit par l'ACP est le plan qui conserve le mieux les proximités entre les individus. Cette notion s'adapte très bien à l'idée de conservation de la proximité entre l'espace des données et l'espace des neurones préconisé par les cartes auto-organisatrices de Kohonen.

L'objectif de cette stratégie est d'initialiser la carte en plaçant les prototypes initiaux sur le plan principal d'inertie, ceci afin de limiter l'ampleur des modifications ainsi que la formation de défauts topologiques de la carte. En effet, une mauvaise initialisation de la carte et un apprentissage mal adapté peuvent conduire à une carte dont la topologie générale respecte peu la distribution des données. L'intérêt de cette méthode est de positionner le plus grand nombre de neurones suivant l'axe portant la plus grande inertie. En d'autres termes, placer d'autant plus de neurones que le nuage de points est allongé suivant un des deux axes principaux. Cette stratégie

d'initialisation par l'ACP a été initialement appliqué dans [37] et [44].

Le nombre initial de neurones sur la grille est défini en fonction des valeurs propres associées aux deux axes formant le plan principal d'inertie. Pour un nombre de neurones fixé a priori, le nombre de neurones de chaque coté d'une grille rectangulaire sera proportionnel à la racine carré de la valeur propre (écart-type) associée à l'axe directeur du même côté. En pratique, il s'agira de résoudre les équations suivantes :

$$\begin{aligned} x_{dim} y_{dim} &= \text{nombre de neurones} \\ \frac{x_{dim}}{y_{dim}} &= \frac{\sqrt{\lambda_1}}{\sqrt{\lambda_2}} \end{aligned}$$

Dans ces équations, x_{dim} et y_{dim} sont, respectivement, les valeurs approchées de la longueur et la largeur de la carte en termes du nombre de neurones, λ_1 et λ_2 sont respectivement la première et la deuxième valeur propre. La figure 2.2 extraite de [44] montre un exemple d'initialisation de la carte de kohonen à partir de la projection du premier plan d'une ACP.

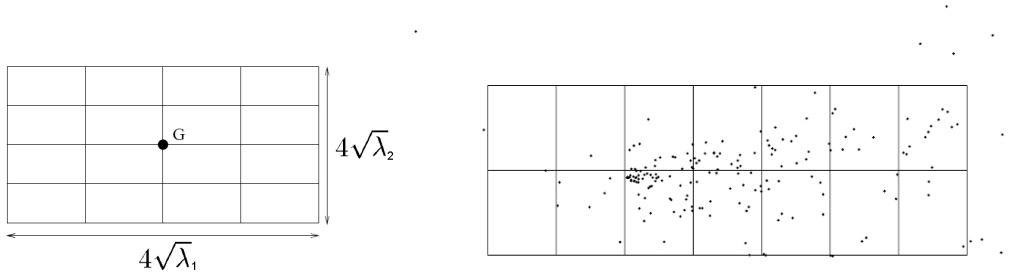

Figure 2.2 – (a) Création d'une grille de longueur $\sqrt{\lambda_1}$ et de largeur $\sqrt{\lambda_2}$ sur le plan formé par les deux premiers vecteurs propres centré sur l'origine du nouveau repère ; (b) Exemple d'une grille sur les points projetés.

4 Nombre de classes à retenir

La recherche du nombre approprié de classes dans une partition est une phase indispensable dans la construction d'une classification de données, mais elle est souvent laborieuse et délicate. Si le choix du nombre de clusters dans une partition ne correspond pas au vrai nombre de classes existantes dans un ensemble d'individus, alors la partition choisie soit regroupe à tort des classes séparées (pas assez de

clusters), soit découpe en plusieurs clusters des classes homogènes (trop de clusters). La recherche du bon nombre de classes se fait par essais et erreurs. Une même technique est utilisée à plusieurs reprises avec un nombre croissant de classes, et pour chaque nouvelle partition obtenue, on calcule la valeur d'un critère de qualité. Le nombre de classes retenu est celui qui conduit à la meilleure valeur de ce critère. Dans la section 4.1, nous décrivons les indices les plus répandus pour la détermination du bon nombre de classes.

On peut également utiliser des aides graphiques. Si la valeur du critère évolue de façon monotone avec le nombre de classes et tend vers une asymptote horizontale lorsque le nombre de classes augmente, on cherche dans la courbe représentant l'évolution de cette quantité un *coude* après lequel son évolution devient négligeable. Le nombre de classes retenu est alors celui correspondant au coude identifié. Cette stratégie pour la détermination du bon nombre de classes est décrite dans la section 4.2.

4.1 Indices de détermination du nombre de classes

La pseudo-statistique de Calinski et Harabasz (1975) [22] est définie selon l'équation 2.11. La valeur de k qui maximise $CH(k)$ est choisie comme le bon nombre de classes.

$$CH(k) = \frac{B/(k-1)}{W/(n-k)} \tag{2.11}$$

L'indice de Baker et Hubert (1975) [10] correspond à l'adaptation de la statistique *Gamma* proposée par Goodman et Kruskal [51] pour le cas d'une partition et est défini selon l'équation 2.12.

Pour calculer cet indice, il faut examiner tous les quadruplets d'individus, et les distances u induites par la partition. Si deux individus i et j appartiennent à la même classe de la partition, alors $u(i,j) = 0$ sinon $u(i,j) = 1$.

Un quadruplet est dit cohérent si $u(i,j) < u(k,l)$ et $d(i,j) < d(k,l)$. Il est dit incohérent si $[u(i,j) < u(k,l)$ et $d(i,j) > d(k,l)]$ ou $[u(i,j) > u(k,l)$ et $d(i,j) < d(k,l)]$. En d'autres termes, un quadruplet est dit cohérent (respectivement, incohérent) si la distance entre deux individus appartenant à une même classe est strictement plus petite (respectivement, grande) que la distance entre deux individus appartenant à de classes différentes.

$$BH(k) = \frac{s(+) - s(-)}{s(+) + s(-)} \tag{2.12}$$

Dans l'équation 2.12, $s(+)$ correspond au nombre de quadruplets cohérents, alors que $s(-)$ correspond au nombre de quadruplets incohérents. Nous avons $BH(k) \in [-1, +1]$. La valeur de k qui maximise $BH(k)$ est choisie comme le bon nombre de classes.

L'indice de Hubert et Levin (1976) [54] est défini selon l'équation 2.13. Dans cette équation, D_{intra} correspond à la somme des distances intra-classe et r est le nombre de distances intra-classe.

$$HL(k) = \frac{D_{intra} - r \min(D_{intra})}{r \max(D_{intra}) - r \min(D_{intra})} \tag{2.13}$$

Nous avons $HL(k) \in [0, 1]$. La valeur de k qui minimise $HL(k)$ est choisie comme le bon nombre de classes. Cet indice est également connu sous le nom de *C-index*.

L'indice Silhouette de Rousseeuw (1987) [84] est défini selon l'équation 2.14. Dans cette équation, $a(i)$ représente la moyenne des distances entre l'individu i et tous les autres individus appartenant à la même classe que l'individu i, $b(i)$ représente la moyenne des distances entre l'individu i et tous les individus appartenant à la classe la plus proche de la classe de l'individu i.

$$S(k) = \frac{1}{n} \sum_{i=1}^{n} s(i) \text{ où } s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{2.14}$$

Nous avons $S(k) \in [-1, +1]$. La valeur de k qui maximise $S(k)$ est choisie comme le bon nombre de classes.

L'indice de Davies et Bouldin (1979) [36] est défini selon l'équation 2.15 en fonction du rapport entre la somme des moyennes des distances intra-classe et la distance entre les centres des classes. Dans cette équation, D_i correspond à la moyenne des distances entre les individus de la classe i. La valeur de k qui minimise l'indice de Davies-Bouldin représente le nombre de classes à retenir pour un bon groupement.

$$DB(k) = \frac{1}{k} \sum_{i,j=1}^{k} \max_{i \neq j} \frac{D_i + D_j}{d(g_i, g_j)} \tag{2.15}$$

L'indice de Krzanowski et Lai (1988) [60] est défini selon l'équation 2.16. La valeur de k qui maximise $KL(k)$ est choisie comme le bon nombre de classes. Dans cette équation, p représente le nombre de variables.

$$KL(k) = \left| \frac{DIFF(k)}{DIFF(k+1)} \right| \text{ où } DIFF(k) = (k-1)^{2/p} W_{k-1} - k^{2/p} W_k \tag{2.16}$$

L'indice de Hartigan (1975) [52] est défini selon l'équation 2.17. $H(k)$ est une F-statistique partielle permettant de tester s'il est intéressant d'ajouter un cluster aux k déjà existants. Pour la détermination du bon nombre de classes, on doit choisir le premier $k >= 1$ tel que $H(k) <= 10$.

$$H(k) = \left(\frac{W_k}{W_{k+1}} - 1 \right) (n - k - 1) \tag{2.17}$$

La statistique Gap de Tibshirani, Walther et Hastie (2001) [95] est calculée comme suit :
Pour c dans l'intervalle $[1, K]$

- Calculer la matrice de dispersion intra-cluster W_c et son déterminant $det(W_c)$
- Construire la courbe des $log[det(W_c)]$ en fonction du nombre c de classes $(c = 1, 2, 3, ..., K)$
- Comparer cette courbe à celle obtenue à partir des données uniformément reparties

L'estimation du nombre de classes à retenir est la valeur c correspondant au plus grand écart entre les deux courbes, ce qui représente la statistique Gap.

4.2 Détermination du nombre de classes par la coupure du dendrogramme

Dans les méthodes de classification hiérarchiques, le dendrogramme indique l'ordre dans lequel les agrégations (ou divisions) successives ont été opérées ainsi que la valeur de l'indice à chaque niveau d'agrégation (ou division). La problématique consiste à trouver le meilleur couple de contraintes opposées : inertie intra-classe et nombre de classes dans la partition, puisque à chaque fois que l'on diminue le nombre de classes dans la partition, son inertie intra-classe augmente.

Il est pertinent d'effectuer la coupure du dendrogramme après les agrégations correspondant à des valeurs peu élevées de l'indice utilisé et avant les agrégations correspondant à des valeurs élevées de cet indice. En coupant le dendrogramme au niveau d'un saut important de cet indice, on peut espérer obtenir une partition de bonne qualité car les individus regroupés en-dessous de la coupure seront proches, et ceux regroupés au-dessus de la coupure seront éloignés.

D'après application de l'algorithme de CAH avec le critère d'agrégation de Ward, on connait à chaque étape les deux classes à rassembler et l'inertie intra-classe de la partition ainsi composée. La meilleure coupure du dendrogramme sera obtenue par un *coude* sur les valeurs du critère optimisé. Ce critère est l'inertie intra-classe. L'on trace le graphique des $n-1$ gains d'inertie intra-classe obtenus à chaque itération de l'algorithme (cf. équation 2.2). Ce coude peut être repéré à l'aide des dérivées (premières et secondes) des valeurs de ce graphique. La stratégie pour la détermination du bon nombre de classes ici décrite est utilisée par le logiciel SPAD [1] [64].

Définition des dérivées première et seconde discrètes

Soit f une fonction et δx une quantité réelle supposée petite. Alors la dérivée première discrète en un point x est définie par l'équation 2.18.

$$Df(x) = \frac{f(x+\delta x) - f(x)}{\delta x} \tag{2.18}$$

La dérivée seconde discrète en un point x est définie par l'équation 2.19.

$$D^2 f(x) = \frac{f(x+\delta x) + f(x-\delta x) - 2f(x)}{\delta x^2} \tag{2.19}$$

Lorsque $\delta x \longrightarrow 0$, on retrouve la dérivée usuelle, définie par la valeur limite du rapport ci-dessus. Dans notre cas, δx est égal à 1. La dérivée première discrète correspond ainsi à la différence première et la dérivée seconde discrète correspond à la différence seconde des $f(x)$.

Définition de la stratégie

Considérons initialement un ensemble de données contenant 5 classes artificiellement générées. Après l'application de l'algorithme de CAH avec la méthode d'agré-

1. `http://www.spadsoft.com/`

gation de Ward, les différences premières et les différences secondes des inerties intra-classes des partitions contenant de 2 à 30 classes sont tracées sur la figure 2.3.

Le bon nombre de classes à retenir sera représenté par la première valeur significative repérée sur le graphique des différences secondes. A partir de l'analyse des graphiques de la figure 2.3, il est facile de vérifier que la meilleure coupure du dendrogramme correspond à une partition en 5 classes (repérée par le pic sur le graphique des différences secondes).

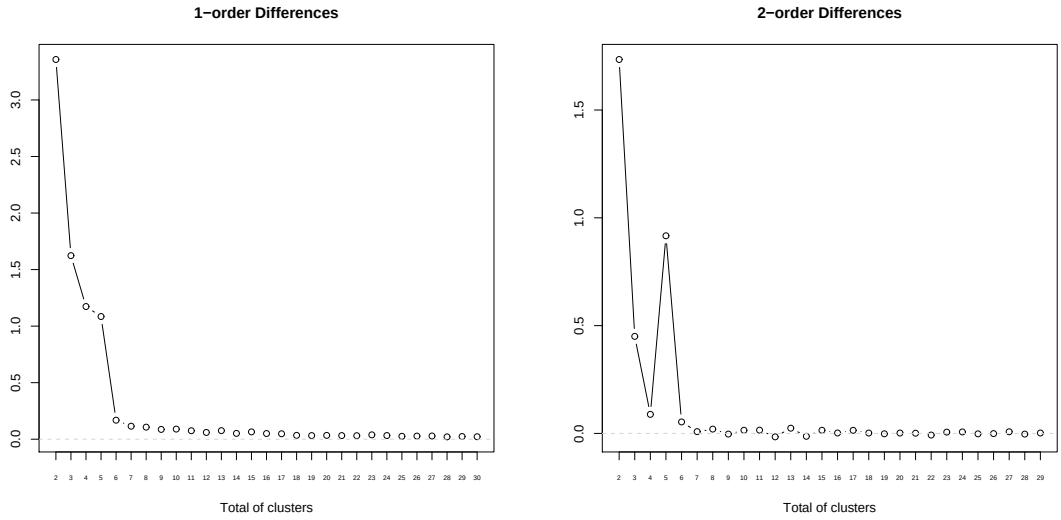

Figure 2.3 – Différences d'inerties intra-classe des partitions obtenues par la CAH appliquant le critère de Ward sur un jeu de données artificielles : premier ordre (à gauche) et second ordre (à droite).

Délibérément dans cet exemple, le jeu de données en question ne contient que des classes bien séparées. Pour le cas des données réelles, le choix du nombre de classes à retenir s'avère plus délicat une fois que plusieurs solutions acceptables peuvent exister, ceci dû notamment aux individus positionnés aux frontières des classes.

Considérons maintenant l'exemple de la figure 2.4 exhibant les différences premières et secondes d'inerties intra-classe issues de la classification sur un jeu de données réelles. Sur le graphique des différences secondes, nous pouvons remarquer l'existence de plusieurs valeurs significatives, ce qui implique plusieurs solutions acceptables.

Une stratégie pour le choix d'une de ces valeurs pourrait être mise en place afin d'automatiser le processus de détermination du bon nombre de classes. Dans le

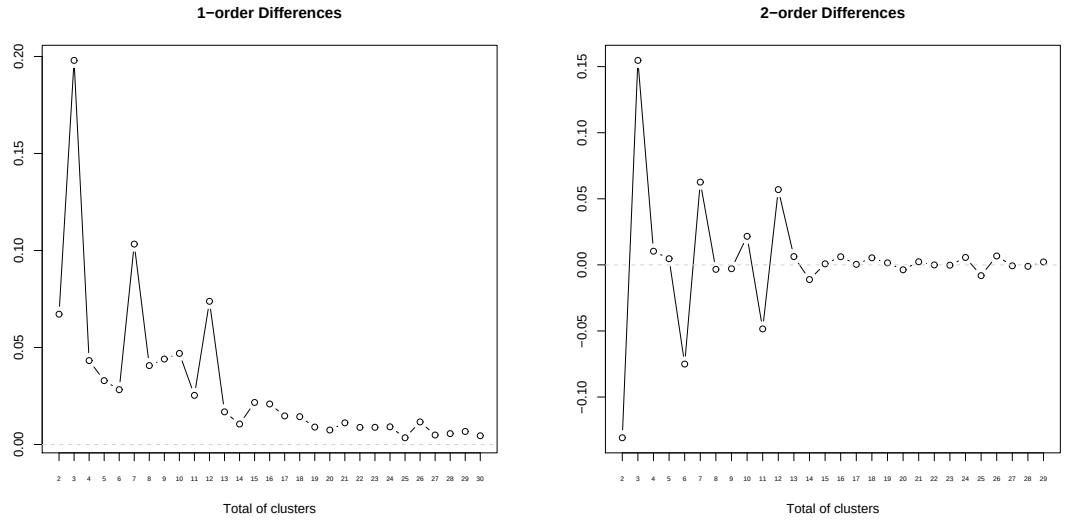

Figure 2.4 – Différences d'inerties intra-classe des partitions obtenues par la CAH appliquant le critère de Ward sur un jeu de données réelles : premier ordre (à gauche) et second ordre (à droite).

cadre de cette thèse, nous en avons expérimenté deux. La première stratégie consiste à parcourir de droite à gauche le graphique des différences secondes et de choisir la première valeur qui est significative au regard des valeurs se trouvant à droite de la valeur analysée. Dans l'exemple de la figure 2.4, le nombre de classes à retenir en appliquant cette stratégie serait 12. La seconde stratégie consiste à parcourir de gauche à droite le graphique des différences secondes et de choisir la première valeur significative, ce qui correspond à chercher la valeur à laquelle correspond le premier pic présent sur ce graphique. Dans l'exemple de la figure 2.4, la valeur choisie en appliquant cette stratégie serait 3.

Comme nous pouvons remarquer, lorsque plusieurs solutions sont acceptables, la première stratégie donne la solution ayant le plus grand nombre de classes. Alors que la seconde stratégie privilégie la solution au plus petit nombre de classes. Pour cette raison, durant nos expérimentations, nous avons opté pour la deuxième stratégie. D'autres stratégies peuvent bien évidement être mises en place.

5 Critères de validation des partitions

Dans cette section, nous décrivons les critères de validation externe adoptés dans notre approche pour la comparaison de deux partitions obtenues sur un même ensemble d'individus. Plus précisément, la comparaison des partitions est basée sur la comparaison des classes qui forment les partitions. La comparaison des classes, quand à elle, est effectuée sur base de leur *extension*, c'est-à-dire, sur l'ensemble d'individus affectés aux classes. Cette stratégie s'oppose aux techniques de comparaison de partition basées sur l'*intension*, c'est-à-dire, sur l'ensemble de représentants (réels ou fictifs) des classes.

5.1 Indices de validation externe basés sur l'extension

Considérons P_1 et P_2 les partitions résultantes de deux classifications sur un même ensemble d'individus E. Le tableau 2.1 représente la matrice de contingence sur ces deux partitions. La valeur n_{ij} dans chaque cellule de ce tableau indique le nombre d'individus présents simultanément dans le cluster C_i de la partition P_1 et dans le cluster C_j de la partition P_2. La partition P_1 est considérée la partition de référence.

	clusters de P_2					
clusters de P_1	C_1	$\cdots$	C_j	$\cdots$	C_k	
C_1	n_{11}	$\cdots$	n_{1j}	$\cdots$	n_{1k}	$n_{1.}$
$\vdots$						
C_i	n_{i1}	$\cdots$	n_{ij}	$\cdots$	n_{ik}	$n_{i.}$
$\vdots$						
C_m	n_{m1}	$\cdots$	n_{mj}	$\cdots$	n_{mk}	$n_{m.}$
	$n_{.1}$		$n_{.j}$		$n_{.k}$	$n_{..} = n$

Table 2.1 – Tableau de contingence entre deux partitions P_1 et P_2.

F-measure

A partir de la matrice de contingence du tableau 2.1, les concepts de *Rappel* et *Précision* peuvent être définis par l'équation 2.20. Le rappel est défini par le ratio entre le nombre d'individus en commun entre deux clusters de différentes partitions et le total d'individus dans le cluster de la partition P_1. La précision est définie par

le ratio entre le nombre d'individus en commun entre deux clusters de différentes partitions et le total d'individus dans le cluster de la partition P_2.

$$rappel(C_i, C_j) = \frac{n_{ij}}{\sum_{j=1}^{k} n_{ij}} \text{ et } precision(C_i, C_j) = \frac{n_{ij}}{\sum_{i=1}^{m} n_{ij}} \tag{2.20}$$

La F-measure [97] combine les concepts de Rappel et Précision dans une seule formule (cf. équation 2.21). Il s'agit d'un cas particulier de la mesure générale F_β (équation 2.22). Pour $\beta = 1$, la F-measure correspond à la moyenne harmonique entre Rappel et Précision.

$$Fmeasure(C_i, C_j) = \frac{2precision(C_i, C_j)rappel(C_i, C_j)}{precision(C_i, C_j) + rappel(C_i, C_j)} \tag{2.21}$$

$$Fmeasure_\beta(C_i, C_j) = \frac{(1+\beta^2)precision(C_i, C_j)rappel(C_i, C_j)}{\beta^2 precision(C_i, C_j) + rappel(C_i, C_j)} \tag{2.22}$$

La F-measure cherche le meilleur représentant d'un cluster A dans la partition P_1 par un cluster A' dans la partition P_2. Si $rappel = precision = 1$, cela signifie $A = A'$. Plus les valeurs du Rappel et de la Précision sont elevées, plus sont semblables les deux clusters. D'une part, si la valeur du Rappel est petite (respectivement, grande), alors cela signifie que le cluster A de la partition P_1 a perdu beaucoup (respectivement, peu) d'individus vis-à-vis de son cluster correspondant dans la partition P_2. D'autre part, si la valeur de la Précision est petite (respectivement, grande), alors cela signifie que le cluster A a gagné beaucoup (respectivement, peu) d'individus au regard de son cluster correspondant dans la partition P_1.

Au niveau de la partition, la F-measure est calculée à partir des moyennes des valeurs maximales sur les colonnes du tableau de contingence (cf. équation 2.23). La F-measure assume des valeurs dans le rang $[0, 1]$ et a pour but fournir une évaluation numérique de la similarité entre deux partitions issues d'un même ensemble d'individus. Plus élevée est la valeur de la F-measure, plus sont similaires les deux partitions.

$$F(P_1, P_2) = \frac{1}{n} \sum_{i=1}^{m} n_{i.} \max_{j=1,\ldots,k} Fmeasure(C_i, C_j) \tag{2.23}$$

Indice de Rand

A partir du tableau de contigence 2.1, il est possible de déduire les formules suivantes :

$$\begin{aligned}
\binom{n_{ij}}{2} &= \frac{n_{ij}(n_{ij}-1)}{2} \text{ représente le nombre de paires d'individus dans } C_i \cap C_j \\
\binom{n}{2} &= \frac{n(n-1)}{2} \text{ représente le nombre de paires de } E \\
\binom{n_{i.}}{2} &= \frac{n_{i.}(n_{i.}-1)}{2} \text{ représente le nombre de paires d'individus du cluster } C_i \\
\binom{n_{.j}}{2} &= \frac{n_{.j}(n_{.j}-1)}{2} \text{ représente le nombre de paires d'individus du cluster } C_j
\end{aligned}$$

On peut encore déduire les formules suivantes :

$$\begin{aligned}
a &= \sum_{i=1}^{m} \sum_{j=1}^{k} \binom{n_{ij}}{2} \\
b &= \sum_{i=1}^{m} \binom{n_{i.}}{2} - \sum_{i=1}^{m} \sum_{j=1}^{k} \binom{n_{ij}}{2} \\
c &= \sum_{j=1}^{k} \binom{n_{.j}}{2} - \sum_{i=1}^{m} \sum_{j=1}^{k} \binom{n_{ij}}{2} \\
d &= \binom{n}{2} + \sum_{i=1}^{m} \sum_{j=1}^{k} \binom{n_{ij}}{2} - \sum_{i=1}^{m} \binom{n_{i.}}{2} - \sum_{j=1}^{k} \binom{n_{.j}}{2}
\end{aligned}$$

Sur ces formules, a représente le nombre de paires d'individus classés ensemble pour chacune des partitions P_1 et P_2, b représente le nombre de paires d'individus classés ensemble selon P_1 et dans deux clusters differents selon P_2, c représente le nombre de paires d'individus classés ensemble selon P_2 et dans deux clusters differents selon P_1 et d représente le nombre de paires d'individus classés dans deux

clusters différents selon P_1 et P_2.

Ces formules sont utilisées dans la définition de l'indice de Rand [82] comme le montre l'équation 2.24.

$$Rand(P_1, P_2) = \frac{a+d}{a+b+c+d} = \frac{\binom{n}{2} + 2\sum_{i=1}^{m}\sum_{j=1}^{k}\binom{n_{ij}}{2} - [\sum_{i=1}^{m}\binom{n_{i.}}{2} + \sum_{j=1}^{k}\binom{n_{.j}}{2}]}{\binom{n}{2}} \quad (2.24)$$

L'indice *Rand* mesure le pourcentage de concordance entre les partitions P_1 et P_2. Il est symétrique, c'est-à-dire, $Rand(P_1, P_2) = Rand(P_2, P_1)$. Cet indice prend des valeurs dans l'intervalle $[0, 1]$. Si les deux partitions comparées sont identiques, alors $Rand(P_1, P_2) = 1$. L'indice *Rand* est d'espérance nulle lorsque les concordances entre les deux partitions sont dues au hasard [101,28,74]. Il correspond ainsi à l'indice *Rand* auquel on apporte une correction pour le manque de valeur constant de *Rand* quand les partitions sont sélectionnées au hasard. Ce nouvel indice, proposé par [53], est connu sous le nom d'indice de Rand corrigé (RC) et défini selon l'équation 2.25.

$$RC(P_1, P_2) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{k}\binom{n_{ij}}{2} - \binom{n}{2}^{-1}\sum_{i=1}^{m}\binom{n_{i.}}{2}\sum_{j=1}^{k}\binom{n_{.j}}{2}}{\frac{1}{2}[\sum_{i=1}^{m}\binom{n_{i.}}{2} + \sum_{j=1}^{k}\binom{n_{.j}}{2}] - \binom{n}{2}^{-1}\sum_{i=1}^{m}\binom{n_{i.}}{2}\sum_{j=1}^{k}\binom{n_{.j}}{2}} \quad (2.25)$$

L'indice RC assume des valeurs contenues dans l'intervalle $[-1, +1]$. Ainsi comme la F-measure, les valeurs proches de 1 correspondent à des partitions très semblables. Les valeurs proches de 0 sont vérifiées lorsque les accords entre les deux partitions sont dûs au hasard, et les valeurs négatives lorsque les partitions sont peu liées.

6 Interprétation d'une partition

L'objectif de cette section est de présenter les critères que nous avons adoptés pour l'interprétation des clusters découverts par la méthode de classification.

Pour la description des données dans d'un un format plus riche, l'Analyse de Données Symboliques (ADS) offre un certain nombre de solutions permettant d'exprimer la variation interne de données à structure complexe (voir section 6.4 du

chapitre 1 pour plus de détails) ainsi que de les interpréter à partir d'un plus haut niveau d'abstraction.

Dans le cadre de cette thèse, nous avons fait le choix de nous limiter à l'interprétation de clusters utilisant la notion de décomposition d'inertie proposée par [25]. Notre but est de pouvoir repérer les variables les plus discriminantes dans la construction de chaque cluster.

6.1 Décomposition de l'inertie sur les classes et les variables

Les inerties peuvent être décomposées en fonction des classes qui constituent la partition ou encore en fonction des variables qui décrivent les individus.

- **L'inertie totale** T sur l'ensemble d'individus E peut s'écrire comme suit :
  $T = \sum_{l=1}^{K} \sum_{j=1}^{p} T_l^j$ avec $T_l^j = \sum_{i \in C_l} w_i d^2(x_i^j, g^j)$
  T_l^j correspond à l'écart moyen, au niveau de la variable j, des individus de la classe C_l au centre de gravité du nuage de points.
- **L'inertie intra-classe** W associée à la partition $P = (C_1, ..., C_l, ..., C_K)$ peut s'écrire comme suit :
  $W = \sum_{l=1}^{K} \sum_{j=1}^{p} W_l^j$ avec $W_l^j = \sum_{i \in C_l} w_i d^2(x_i^j, g_l^j)$
  W_l^j correspond à l'inertie intra-classe de la classe C_l au niveau de la variable j.
- **L'inertie inter-classe** B associée à la partition $P = (C_1, ..., C_l, ..., C_K)$ peut s'écrire comme suit :
  $B = \sum_{l=1}^{K} \sum_{j=1}^{p} B_l^j$ avec $B_l^j = \mu_l d^2(g_l^j, g^j)$
  B_l^j correspond à l'écart, au niveau de la variable j, du centre de gravité de la classe C_l au centre de gravité du nuage de points.

Par analogie, il est possible de décomposer les inerties sur les classes et sur les variables séparément, comme le montre les paragraphes suivants.

Décomposition des inerties sur les classes

$$\begin{aligned} T_l &= \sum_{i \in C_l} w_i d^2(x_i, g) \\ W_l &= \sum_{i \in C_l} w_i d^2(x_i, g_l) \\ B_l &= \mu_l d^2(g_l, g) \end{aligned}$$

Décomposition des inerties sur les variables

$$\begin{aligned} T^j &= \sum_{i=1}^{n} w_i d^2(x_i^j, g^j) \\ W^j &= \sum_{l=1}^{K} \sum_{i \in C_l} w_i d^2(x_i^j, g^j) \\ B^j &= \sum_{l=1}^{K} \mu_l d^2(g_l^j, g^j) \end{aligned}$$

Ces décompositions de l'inertie peuvent être liées de façon additive par la relation fondamentale $T = W + B$, comme suit :

$$\begin{aligned} \sum_{l=1}^{K} \sum_{j=1}^{p} T_l^j &= \sum_{l=1}^{K} \sum_{j=1}^{p} W_l^j + \sum_{l=1}^{K} \sum_{j=1}^{p} B_l^j \\ \sum_{l=1}^{K} T_l &= \sum_{l=1}^{K} W_l + \sum_{l=1}^{K} B_l \\ \sum_{j=1}^{p} T^j &= \sum_{j=1}^{p} W^j + \sum_{j=1}^{p} B^j \end{aligned}$$

6.2 Contributions des variables

La part de l'inertie expliquée en groupant les individus de E en une partition $P = (C_1, ..., C_K)$ s'écrit selon l'équation 2.26. Plus R est fort, plus la partition constitue une bonne représentation des individus. Si $R = 1$, le nuage d'individus correspond aux K prototypes des classes.

$$R = \frac{B}{T} \tag{2.26}$$

Pour chaque variable j, un indice analogue à R est définit (cf. équation 2.27). Cet indice représente la part d'inertie de la variable j pris en compte par la partition P. Il mesure ainsi le pouvoir discriminant de la variable j par rapport à la partition P.

$$COR(j) = \frac{B^j}{T^j} \tag{2.27}$$

En comparant cet indice avec R, nous avons :

- Si $COR(j) > R$, la variable j est plus discriminante que la moyenne puisque R représente le pouvoir discriminant moyen vis-à-vis de la partition.
- Si $COR(j) < R$, la variable j est moins discriminante que la moyenne.

La contribution relative de la variable j à l'inertie inter-classe de la partition P est décrite selon l'équation 2.28.

$$CRT(j) = \frac{B^j}{B} \text{ avec } \sum_{j=1}^{p} CRT(j) = 1 \tag{2.28}$$

6.3 Interprétation statistique des classes par variable

Nous nous concentrons maintenant sur les indices d'interprétation des classes par un ensemble de variables [25]. Ces indices sont fondés sur la décomposition de l'inertie associée à un ensemble d'individus et à la partition dans laquelle ceux-ci sont classés. La stratégie d'interprétation des classes par des variables consiste à calculer, pour chacune des variables, un critère mensurant sa pertinence dans l'interprétation de la classe. Pour chaque variable j et chaque classe C_l, les critères suivants sont définis :

- **Le pouvoir discriminant de la variable j par rapport à la classe C_l :**

$$COR(j,l) = \frac{B_l^j}{T^j} \text{ avec } COR(j) = \sum_{l=1}^{K} COR(j,l) \tag{2.29}$$

  Cet indice représente le pourcentage du pouvoir discriminant de la variable j pris en compte par la classe C_l. Plus cet indice est fort, plus la variable j aura un comportement homogène sur les individus de la classe C_l. Examiner cet indice est particulierement intéressant pour les variables à fort pouvoir discriminant, il indique la répartition entre les classes de ce pouvoir discriminant.
- **La contribution relative de la variable j à l'interie inter-classe du cluster C_l :**

$$CTR(j,l) = \frac{B_l^j}{B_l} \tag{2.30}$$

Cet indice est complémentaire au précédent et permet de determiner les variables qui caracterisent le plus chaque cluster.

7 Synthèse

Au cours de ce chapitre, nous avons présenté les méthodes de classification automatique faisant objet des expérimentations réalisées dans le cadre de cette thèse. Si bien que l'approche proposée pour la détection et le suivi des changements (cf. section 3 du chapitre 3) soit totalement indépendante de la méthode de classification appliquée, nous avons voulu l'expérimenter avec les méthodes de classification les plus répandues. Dans la section 4, nous avons abordé la problématique de détermination du bon nombre de classes au travers la présentation de quelques indices et stratégies permettant de trouver une solution approchée. Dans la section 5 du présent chapitre, nous avons présenté deux indices de validation externe pour la comparation de deux partitions obtenues sur un même ensemble d'individus. Finalement, dans la section 6, nous avons présenté les indices adoptés dans notre approche pour l'interprétation des classes d'une partition selon les variables les plus discriminantes.

Chapitre 3

Contributions

1 Introduction

Ce chapitre a pour objectif de décrire les contributions nées des travaux de recherche réalisés dans le cadre de cette thèse.

La première contribution concerne la modélisation des données d'usage. Dans un premier temps, nous proposons un ensemble de variables statistiques ayant pour but de caractériser la manière dont un internaute navigue sur un site Web. Dans un deuxième temps, nous proposons la modélisation des activités des internautes à l'aide d'un tableau de comptage des clics effectués sur les thèmes de pages du site Web visité. Les détails de ces deux propositions sont présentés dans la section 2.

Pour pallier le problème d'acquisition des données réelles d'usage, nous proposons une méthodologie pour la génération automatique des données artificielles permettant également la simulation des changements. Cette méthodologie est détaillée dans la section 3.

Ayant pour but de trouver une solution pour le problème de détection de changements sur les données évolutives, nous avons mis en place quelques analyses exploratoires. Ces analyses concernent différentes stratégies de classification présentées dans la section section 4 du présent chapitre. Guidés par les pistes nées de ces analyses préliminaires, nous proposons dans la section 5 une nouvelle approche pour la détection et le suivi des changements sur des données évolutives.

2 Modélisation des données d'usage

2.1 Variables statistiques pour la caractérisation des navigations sur un site Web

Dans le but de mieux caractériser la navigation d'un internaute, nous définissons un ensemble de 14 variables statistiques obtenues à partir des données d'usage prétraitées et stockées dans un entrepôt (voir la section 4 du chapitre 4 pour la spécification des commandes SQL utilisées pour extraire ces variables à partir des tables d'une base de données relationnelle). Ces variables sont énumérées dans le tableau 2.1 et détaillées dans les paragraphes suivants.

N^o	Variable	Signification
1	NbRequests	Nombre de clics effectués durant la navigation
2	NbRequests_OK	Nombre de requêtes réussies (statut = 200) dans la navigation
3	NbRequests_Bad	Nombre de requêtes échouées (statut $\neq$ 200) dans la navigation
4	PRequests_OK	Pourcentage de requêtes réussies (= $NbRequests\_OK/NbRequests$)
5	NbRepetitions	Nombre de requêtes répétées dans la navigation
6	PRepetitions	Pourcentage de répétitions (= $NbRepetitions/NbRequests$)
7	DureeTotale	Durée totale de la navigation (en secondes)
8	MDuree	Moyenne de la durée des requêtes (= $DureeTotale/NbRequests$)
9	MDuree_OK	Moyenne de la durée des requêtes réussies (= $DureeTotale\_OK/NbRequests\_OK$)
10	NbRequests_Sem	Nombre de requêtes liées aux pages de la structure sémantique du site
11	PRequests_Sem	Pourcentage de requêtes liées aux pages de la structure sémantique du site (= $NbRequests\_Sem/NbRequests$)
12	TotalSize	Total d'octets transférés durant la navigation
13	MSize	Moyenne d'octets transférés durant la navigation (= $TotalSize/NbRequests\_OK$)
14	DureeMax_OK	Durée maximale parmi les durées des requêtes réussies

Table 3.1 – Variables statistiques décrivant les navigations.

Durant la navigation sur un site Web, l'utilisateur peut faire face à des messages d'erreur sur la page demandée (quand la page n'a pas été retrouvée), de redirection (quand la page requise a été physiquement déplacée), d'erreur sur le serveur Web (quand le serveur en question est occupé ou hors service), etc. Le code de retour d'une requête est identifié dans les fichiers log Web par le champ *statut*. Les codes de retour de requête assument des valeurs standardisées par le W3C [1]. Une requête est dite réussie quand son code de retour est égal à 200. Dans tous les autres cas, la requête est considérée échouée. Dans le but de discriminer le nombre de requêtes réussies et échouées durant une navigation, nous utilisons respectivement les va-

1. Une liste exhaustive des codes de retour de requête peut être consultée à l'adresse suivante : `http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html`

riables $NbRequests\_OK$ et $NbRequests\_Bad$. La variable $NbRequests$ représente le nombre total de clics effectués pendant la navigation, tous statuts confondus.

Durant la navigation, un internaute peut, pour un motif quelconque, revenir sur une page déjà visitée. Les pages revisitées durant une navigation sont souvent celles qui attirent le plus l'attention de l'internaute. Les variables $NbRepetitions$ et $PRepetitions$ contiennent respectivement le nombre et le pourcentage de pages revisitées dans une navigation.

La variable $DureeTotale$ représente le temps (en secondes) consacré par l'internaute à la visite du site Web en question. La variable $MDuree$ représente la moyenne de temps consacré aux pages visitées durant la navigation. La variable $MDuree\_OK$ continent la moyenne de temps de visite parmi les requêtes réussies.

Si les pages principales du site en question sont organisées sous une structure sémantique préétablie, les variables 10 et 11 sont prévues pour discriminer les clics sur les pages de cette structure. La variable $NbRequests\_Sem$ compte le nombre de clics réalisés par l'internaute sur les pages présentes dans la structure sémantique. La variable $PRequests\_Sem$ représente le pourcentage de clics consacrés aux pages de la structure sémantique parmi tous les clics réalisés durant la navigation.

La variable $TotalSize$ compte le nombre d'octets transmis par le serveur Web à la machine de l'internaute durant toute sa navigation. La variable $MSize$ représente la moyenne de ces valeurs. La variable $DureeMax\_OK$ continent la durée maximale (en secondes) consacrée par l'utilisateur pour la visualisation d'une page.

Les variables présentées dans cette section ont pour but de décrire la navigation des internautes d'un site Web quelconque à partir des données extraites des fichiers log du serveur Web qui héberge le site en question. Elles peuvent ainsi contribuer à une analyse plus fine des habitudes de navigation présentées par le public du site Web auditionné.

2.2 Tableau de comptage des clics sur les thèmes de pages d'un site Web

Dans le but d'identifier la distribution des clics d'un internaute sur l'ensemble de pages d'un site Web, nous proposons la modélisation des données d'usage sous forme d'un tableau de comptage. Le nombre de pages d'un site Web pouvant atteint rapidement de grandes dimensions, il est envisageable de réaliser une segmentation préalable des pages par thèmes. Le tableau de comptage proposé est illustré par le

tableau 3.2.

ID	thème$_1$	thème$_2$	$\cdots$	thème$_j$	$\cdots$	thème$_{p-1}$	thème$_p$	date-heure
1	C_1^1	C_1^2	$\cdots$	$\vdots$	$\cdots$	C_1^{p-1}	C_1^p	JJ/MM/AAAA hh :mm :ss
2	C_2^1	C_2^2	$\cdots$	$\vdots$	$\cdots$	C_2^{p-1}	C_2^p	JJ/MM/AAAA hh :mm :ss
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	C_i^1	C_i^2	$\cdots$	C_i^j	$\cdots$	C_i^{p-1}	C_i^p	JJ/MM/AAAA hh :mm :ss
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 3.2 – Tableau de comptage pour la modélisation des données d'usage.

Dans le tableau 3.2, chaque ligne représente une navigation d'internaute, la première colonne représente l'identificateur unique de la navigation, la dernière colonne représente la date et l'heure de début de la navigation, ce qui correspond à la date du premier clic effectué par l'internaute durant la navigation en question. Les cellules du type C_i^j de ce tableau représentent le nombre de clics réalisés durant la navigation i sur les pages du thèmes j.

La modélisation des données d'usage par moyen d'un tableau de comptage reste une approche assez générique permettant l'application de méthodes aptes à traiter ce type de données.

3 Méthodologie pour la génération de données artificielles d'usage

Dans cette section, nous allons présenter une méthodologie pour la génération de données d'usage artificielles sous la forme d'un tableau de comptage *navigations* $\times$ *thème de pages* (cf. tableau 3.2). Notons que dans notre approche, l'unité élémentaire est une *navigation* d'internaute et non un clic isolé. Cette procédure est assez éloignée des analyses statistiques élémentaires dans lesquelles on compte les accès à une page en fonction de la date/l'heure, sans tenir compte des navigations. La méthodologie proposée spécifie également des fonctionnalités additionnelles permettant la simulation de changements (cf. sections 3.1 et 3.2) sur les données artificielles générées. Notre principale motivation est la possibilité de mesurer l'efficacité d'une approche de détection de changements sur un ensemble de données contenant des changements contrôlés.

Soit $(x_1, \ldots, x_i, \ldots, x_n)$ la représentation vectorielle de l'ensemble d'individus que nous voudrions générer (dans notre cas, un individu correspond à une navigation). Un individu i est décrit par un ensemble de p variables : $x_i = \left\{x_i^1, \ldots, x_i^j, \ldots, x_i^p\right\}$. La valeur x_i^j représentera le nombre de clics réalisés durant la navigation i sur les pages du thème j. Considérons également $(p_1, \ldots, p_c, \ldots, p_K)$ l'ensemble de prototypes initiaux (noyaux) de K classes *a priori*, où $p_c = \{p_c^1, \ldots, p_c^j, \ldots, p_c^p\}$. Chaque prototype p_c représentera un profil d'usage, où chaque valeur p_c^j représente la moyenne de clics effectués par les individus de la classe c sur les pages du thème j. Soit $(\alpha_1, \ldots, \alpha_c, \ldots, \alpha_k)$ le pourcentage d'effectifs de chacune de ces classes, où $\alpha_c \in [0, 1]$ et $\sum_{c=1}^{k} \alpha_c = 1$. Dans nos algorithmes, le nombre total n d'individus artificiels à générer est représenté par la variable $totalInd$ et doit être spécifié à l'avance.

L'algorithme générique pour la génération des données artificielles à partir des profils initiaux de K classes *a priori* est décrit comme suit :

```
1  Pour c dans l'intervalle[1,K] {
2    Pour chaque individu i de la classe c {
3      x_i ← {0,...,0}
4      nbClic ← random (nbClicMIN, nbClicMAX)
5      Pour y dans l'intervalle[1,nbClic] {
6        z ← random(0,1)
7        Pour j dans l'intervalle[1,p] {
8          Si (z <= cumul(p_c^j)) {
9            x_i^j ← x_i^j + 1
10           Sortir de la boucle la plus interne
11         }
12       }
13     }
14   }
15 }
```

A la ligne 4 de cet algorithme, un entier compris entre $nbClicMIN$ et $nbClicMAX$ est tiré au hasard et attribué à la variable $nbClic$. Cette valeur représente le nombre total de clics effectués durant la navigation i. Une fois le nombre de clics défini, nous devons les distribuer parmi les p thèmes de pages, et ce tout en respectant la distribution des clics imposée par les prototypes des classes *a priori*. Pour ce faire, nous appliquons la fonction *cumul* (ligne 8) sur la fréquence des clics présente dans les prototypes des classes.

Les sections suivantes ont pour but de présenter les fonctionnalités additionnelles

permettant la simulation des changements sur les données artificielles générées. Les changements simulés peuvent être liés à l'effectif et à la position initiale des classes artificielles dans l'espace des données.

3.1 Changement lié à l'effectif des classes

En ce qui concerne les changements liés à l'effectif des classes, nous pouvons en avoir deux types : le *rétrécissement de la classe cible*, provoqué par la baisse de son effectif et le *grossissement de la classe cible*, provoqué par l'augmentation de son effectif. Les changements sur des données artificielles se font par paquet de données, ces paquets sont ici appelés *fenêtres*. Le nombre total de fenêtres à générer doit être fixé à l'avance. Afin de maintenir constant le nombre d'individus dans une même fenêtre, le rétrécissement de la classe cible impliquera le grossissement des autres classes dans la fenêtre. De la même manière, le grossissement de la classe cible impliquera le rétrécissement des autres classes dans la fenêtre en question.

L'intensité des changements se fait à l'aide de deux facteurs β_1 et β_2, où $\beta_1, \beta_2 \in [0, 1]$ et $\beta_1\beta_2 = 1$, définis dans les sections suivantes.

Rétrécissement d'une classe

Pour la simulation de l'effet de rétrécissement d'une classe cible, nous devons multiplier son effectif par le facteur de rétrécissement β_1 où $\beta_1 \in [0, 1]$. Les autres classes doivent avoir leur effectif multiplié par le facteur $\beta_1^{'}$, défini en fonction de β_1 selon l'équation 3.1. Dans cet exemple, la classe cible est numérotée 1.

$$\begin{aligned} \alpha_1 + \alpha_2 + \ldots + \alpha_K &= 1 \\ \beta_1\alpha_1 + \beta_1^{'}\alpha_2 + \ldots + \beta_1^{'}\alpha_K &= 1 \\ \beta_1\alpha_1 + \beta_1^{'}(1 - \alpha_1) &= 1 \\ \beta_1^{'} &= \frac{1 - \beta_1\alpha_1}{1 - \alpha_1} \end{aligned} \tag{3.1}$$

L'algorithme pour le rétrécissement d'une classe cible est décrit comme suit :

```
1 Pour c dans l'intervalle[1, K] {
2   Tirer au hasard et sans remise (α_c totalInd) individus de
3   la classe c et les placer dans la fenêtre W^1
4 }
5 Pour t dans l'intervalle[2, nbWindow] {
6   Pour c dans l'intervalle [1, K]
7     Si (c = c')
8       α_c ← α_c β_1
9     Sinon
10      α_c ← α_c β'_1
11    Tirer au hasard et sans remise (α_c totalInd) individus
12    de la classe c et les placer dans la fenêtre W^t
13  }
14 }
```

Dans cet algorithme, la boucle des lignes 1-4 a pour but de remplir la première fenêtre des données avec un échantillon d'individus de chaque classe. Pour simuler le rétrécissement de la classe cible, nous mettons à jour son effectif à l'aide du facteur de rétrécissement β_1 (ligne 8). Les autres classes ont également leur effectif actualisé à l'aide du facteur $\beta_1^{'}$ (ligne 10). Des nouveaux élements de chaque classe sont alors échantillonnés et placés dans la fenêtre suivante (lignes 11-12).

Grossissement d'une classe

Pour la simulation de l'effet de grossissement d'une classe cible, nous devons diviser son effectif par le facteur de grossissement β_2 où $\beta_2 \in [0, 1]$. Les autres classes doivent avoir leur effectif multiplié par $\beta_2^{'}$, défini en fonction de β_2 selon l'équation 3.2. Dans cet exemple, la classe cible est numérotée 1.

$$\begin{aligned} \alpha_1 + \alpha_2 + \ldots + \alpha_K &= 1 \\ \frac{\alpha_1}{\beta_2} + \beta_2^{'}\alpha_2 + \ldots + \beta_2^{'}\alpha_K &= 1 \\ \frac{\alpha_1}{\beta_2} + \beta_2^{'}(1 - \alpha_1) &= 1 \\ \beta_2^{'} &= \frac{1 - \frac{\alpha_1}{\beta_2}}{1 - \alpha_1}, \text{ où } \beta_2 > \alpha_1 \end{aligned} \tag{3.2}$$

L'algorithme pour le grossissement de la classe cible est décrit comme suit :

```
1 Pour c dans l'intervalle[1,K] {
2   Tirer au hasard et sans remise (α_c totalInd) individus de la
3   classe c et les placer dans la fenêtre W^1
4 }
5 Pour t dans l'intervalle[2,nbWindow] {
6     Pour c dans l'intervalle_[1,K] {
7       Si (c = c')
8         α_c ← α_c/β_2
9       Sinon
10        α_c ← α_c β'_2
11      Tirer au hasard et sans remise (α_c totalInd) individus
12      de la classe c et les placer dans la fenêtre W^t
13    }
14 }
```

Cet algorithme est assez proche de son précédent. La seule différence est présente dans la mise à jour des effectifs des classes, faite à l'aide des facteurs de grossissement β_2 et $\beta_2^{'}$ (lignes 8 et 10).

3.2 Changement lié à la position des classes dans l'espace des données

Soit $(p_1, \ldots, p_c, \ldots, p_K)$ l'ensemble de profils initiaux des K classes *a priori*. Chaque profil de classe est décrit par p variables : $p_c = \{p_c^1, \ldots, p_c^j, \ldots, p_c^p\}$.

Supposons que l'on veuille simuler l'effet de déplacement dans l'espace d'une classe c vers une classe cible $c^{'}$. La figure 3.1 montre un exemple où $K = 5$, $c = 2$ et $c^{'} = 1$. Pour ce faire, nous devons ajouter à chaque variable j de l'individus i appartenant à la classe déplaçante c, un Δ représentant l'écart entre les profils des classes c et $c^{'}$ au regard de la variable j. Cet écart est défini selon l'équation 3.3, où $\beta_3 \in [0, 1]$ est le facteur de déplacement et a pour but de contrôler l'intensité du déplacement des classes. Plus élevée est la valeur de ce facteur, plus proche de la classe cible sera la classe déplacée.

$$\Delta^j = (p_{c'}^j - p_c^j)\beta_3 \tag{3.3}$$

L'algorithme pour le déplacement des classes artificielles vers une classe cible est

décrit comme suit :

```
Pour c dans l'intervalle[1, K] {
  Tirer au hasard et sans remise (α_c totalInd) individus de la
  classe c et les placer dans la fenêtre W^1
}
Pour t dans l'intervalle[2, nbWindow] {
  Pour c dans l'intervalle [1, K] {
    Si (c = c')
      Placer les individus de la classe c dans la fenêtre W_t
    Sinon
      Pour chaque individu i de la classe c {
        Créer un nouvel individu i'
        Pour chaque j dans l'intervalle[1, p] {
          Δ^j ← (p^j_c' − p^j_c)β_3
          x^j_i' ← x^j_i + Δ^j
        }
        Placer l'individu i' dans la fenêtre W^t
      }
    }
  }
}
```

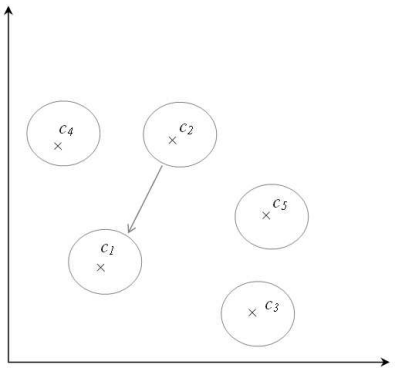

Figure 3.1 – Déplacement de la classe 2 vers la classe 1.

Dans cet algorithme, toutes les classes artificielles, hormis la classe cible, ont leurs individus déplacés. Ceci est fait en créant un nouvel individu $i^{'}$ à partir de chaque individu i appartenant aux classes déplaçantes. La représentation vectorielle de l'individu $i^{'}$ est définie en additionnant, pour chaque variable j, l'écart Δ vérifié entre le profil de la classe à laquelle appartient l'individu i et le profil de la classe

cible (ligne 13). Les nouveaux individus créés sont placés dans la fenêtre suivante (ligne 17).

4 Analyse exploratoire des stratégies de classification

L'analyse proposée dans ce chapitre consiste dans un premier temps à diviser la période analysée en sous périodes de données plus significatives dans le but de rechercher l'apparition et/ou l'évolution des comportements qu'on manquerait par une analyse globale de toute la période. Pour ce faire, nous avons adopté la stratégie d'analyse de données par paquets. Dans le cadre de nos travaux, ces paquets de données sont appelés *fenêtres* (ou *windows* dans le vocabulaire anglophone utilisé dans l'analyse des flux des données). Les fenêtres utilisées dans notre approche sont du type *non recouvrante*, en opposition aux fenêtres du type *glissante*, qui présentent celles-ci une zone de chevauchement.

Concernant le partitionnement des données sur des fenêtres, nous en définissons deux types :

1. **Partitionnement par nombre d'effectifs constant** (fenêtre logique) : on fixe le nombre d'individus qui doivent être contenus dans chaque fenêtre ;
2. **Partitionnement par intervalle de temps constant** (fenêtre temporelle) : on fixe un intervalle de temps durant lequel les données analysées seront enregistrées dans une fenêtre, par exemple 30 minutes, 2 heures, 1 jour, 1 semaine, 1 mois, des périodes dans la journée (matin, après midi, soir), etc. Notons que dans ce cas-ci, le nombre d'individus dans la fenêtre peut varier. Dans le cadre de l'analyse d'usage du Web, si on applique une répartition des données par fenêtres temporelles de taille égale à 1 moins, chaque fenêtre enregistrera les navigations d'internaute ayant lieu de 00 :00 :00 du premier jour du mois jusqu'à 23 :59 :59 du dernier jour d'un mois en question. Par la suite, une nouvelle fenêtre est définie afin d'enregistrer les navigations du mois suivant, et ainsi de suite. Dans cet exemple, le pas de temps t est défini en termes d'unités mensuelles. Ainsi, la fenêtre de temps $t+1$ enregistre les navigations réalisées durant le mois immédiatement suivant au mois de la fenêtre de temps t.

Par la suite, une classification non supervisée est appliquée sur les données de chaque fenêtre (dans les stratégies de classification dites *locales*), aussi bien que sur tout l'ensemble de données (dans la stratégie de classification *globale*). Les résultats fournis pour chaque stratégie de classification sont comparés les uns avec les autres. Notons que dans notre approche, la classification des données portera sur un ensemble d'individus (navigations) qui partagent la même fenêtre (logique ou temporelle). Chaque navigation sera affectée à un groupe (cluster) contenant des navigations proches au sens de la distance appliquée par la méthode de classification.

4.1 Présentation des stratégies de classification

Dans le but de définir une méthode efficace pour la détection des changements entre les données de deux fenêtres voisines, nous avons exploré quatre stratégies de classification non supervisée, décrites dans les sections suivantes.

Classification globale (CG)

Dans cette stratégie, la méthode de classification non supervisée doit être appliquée sur tout l'ensemble d'individus à analyser. Cette approche correspond à la technique traditionnelle de classification où tout le tableau de données associé à l'ensemble E, de taille n, doit être chargé en mémoire.

Par l'application d'un filtre sur chaque cluster obtenu, il est possible de définir une partition pour chaque fenêtre adoptée. En d'autres termes, par l'intersection des clusters avec les fenêtres, la stratégie de classification globale engendre une partition sur chaque fenêtre.

La procédure à exécuter pour cette stratégie de classification peut être décrite comme suit :

- Appliquer la méthode de classification sur l'ensemble de données E avec une initialisation aléatoire des prototypes.
- Pour chaque cluster de la partition obtenue :
  - Partitionner le cluster en appliquant un filtre (logique ou temporel)
  - Positionner les nouveaux clusters résultants dans leurs respectives fenêtres

Les étapes de cette stratégie de classification sont illustrées par la figure 3.2. Sur cette figure, les partitions marquées en rouges sont conservées afin d'être comparées aux partitions obtenues par les autres stratégies de classification appliquées sur le même ensemble d'individus.

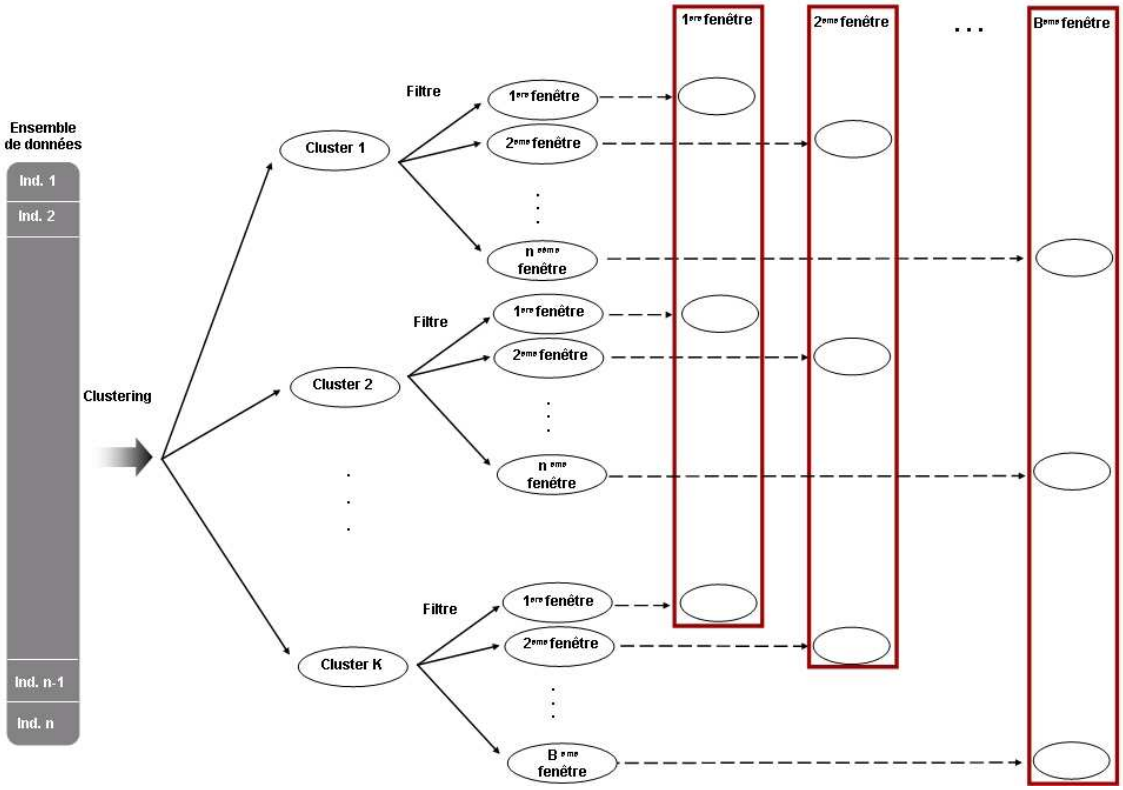

Figure 3.2 – Schéma de la stratégie de classification globale

Classification locale indépendante (CL_1)

Dans cette stratégie, la méthode de classification non supervisée doit être appliquée séparément sur les données de chaque fenêtre. Les partitions obtenues sont ainsi totalement indépendantes les unes des autres.

La procédure à exécuter pour cette stratégie de classification peut être décrite comme suit :

- Partitionner l'ensemble E en B fenêtres (logiques ou temporelles)
- Pour chaque fenêtre
  - Appliquer la méthode de classification avec une initialisation aléatoire des prototypes

Les étapes de cette stratégie de classification sont illustrées sur la figure 3.3. Les partitions obtenues sur chaque fenêtre de données (marquées en rouge sur la figure 3.3) sont conservées afin d'être comparées aux partitions obtenues par les autres stratégies de classification appliquées sur le même ensemble d'individus.

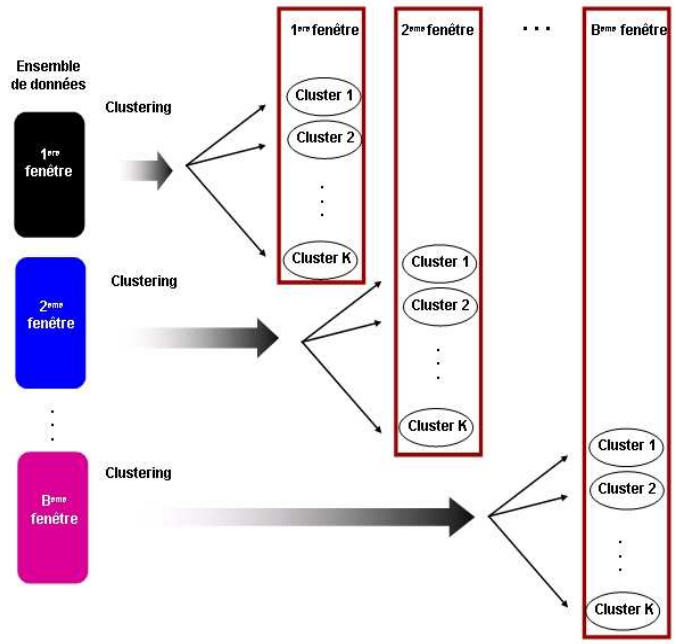

Figure 3.3 – Schéma de la stratégie de classification locale indépendante

Classification locale "précédente" (CL_2)

Dans cette stratégie, la méthode de classification non supervisée doit utiliser la structure classificatoire (prototypes des classes) issue de la fenêtre immédiatement précédente pour obtenir une partition sur les données de la fenêtre courante. En d'autres termes, la méthode de classification doit effectuer une seule étape d'affectation des données de la fenêtre courante aux prototypes issus de la fenêtre immédiatement précédente.

La partition obtenue sur la fenêtre courante reflétera ainsi l'organisation des données de la fenêtre courante selon la structure classificatoire provenant de la fenêtre immédiatement précédente dans le temps.

La procédure à exécuter pour cette stratégie de classification peut être décrite comme suit :

- Partitionner l'ensemble E en B fenêtres (logiques ou temporelles)
- Pour chaque fenêtre et tant qu'il y a des fenêtres
  - Appliquer une méthode fournissant une partition sur les individus de la fenêtre courante
  - Affecter les individus de la fenêtre immédiatement suivante aux prototypes des classes de la fenêtre courante

La méthode appliquée sur chaque fenêtre pour le partitionnement de ses individus est de libre choix. Par exemple, cela pourrait être : (1) une méthode de classification avec une initialisation aléatoire des prototypes, ou (2) une simple étape de mise à jour des prototypes après l'étape d'affectation des individus.

Soit $L_1, L_2, ..., L_k$ l'ensemble des prototypes finaux des classes sur une fenêtre, les étapes de la stratégie de classification précédente sont illustrées par la figure 3.4. Sur cette figure, les partitions marquées en rouge sont conservées afin d'être comparées aux partitions obtenues par les autres stratégies de classification appliquées sur le même ensemble d'individus.

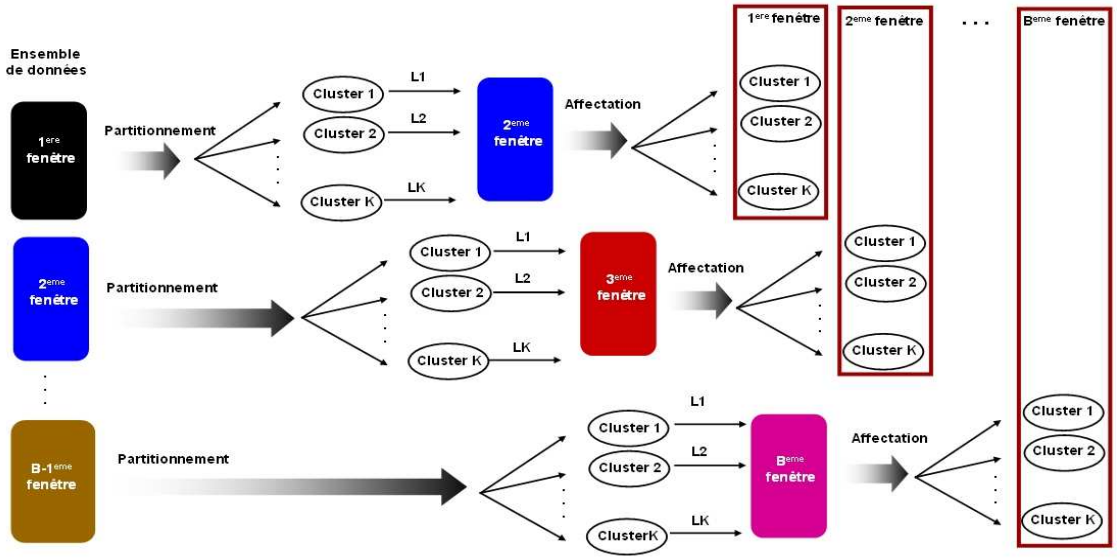

Figure 3.4 – Schéma de la stratégie de classification locale "précédente"

Classification locale dépendante (CL_3)

Dans cette stratégie, la méthode de classification non supervisée doit également utiliser la structure classificatoire issue de la fenêtre immédiatement précédente pour, cette fois-ci, initialiser une nouvelle classification sur les données de la fenêtre courante. En d'autres termes, la méthode de classification sera initialisée à partir des prototypes finaux résultants de la classification réalisée sur la fenêtre immédiatement précédente.

La procédure à exécuter pour cette stratégie de classification peut être décrite comme suit :

– Partitionner l'ensemble E en B fenêtres (logiques ou temporelles)
– Appliquer la méthode de classification sur les données de la première fenêtre avec une initialisation aléatoire des prototypes
– A partir de la deuxième fenêtre et tant qu'il y a des fenêtres :
  • Appliquer la méthode de classification avec une initialisation à partir des prototypes issus de la fenêtre immédiatement précédente

Les étapes de cette stratégie de classification sont illustrées par la figure 3.5 où les partitions marquées en rouge sont également conservées afin d'être comparées aux partitions obtenues par les autres stratégies de classification appliquées sur le même ensemble d'individus.

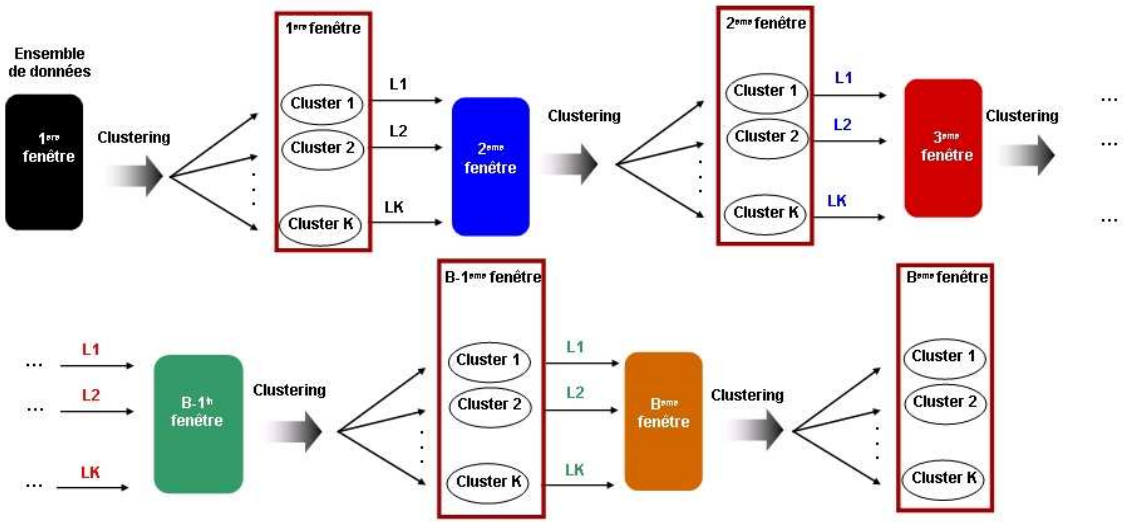

FIGURE 3.5 – Schéma de la stratégie de classification locale dépendante

4.2 Caractéristiques des stratégies de classification

Pour la mise en place de la stratégie de classification globale (CG), tout le tableau de données doit être chargé en mémoire centrale. Dans le cas des grands tableaux de données, cette contrainte ne peut pas toujours être satisfaite en raison des limites physiques des mémoires des machines.

Pour palier ce problème, les stratégies de classification dites *locales* (CL_1, CL_2, CL_3) proposent une solution alternative. Dans ces stratégies, contrairement à la stratégie CG, seul le tableau de données associé à une fenêtre est chargé en mémoire centrale. Ce processus est réalisé de façon séquentielle, c'est-à-dire, quand le

traitement d'une fenêtre de données se termine, la fenêtre en question peut quitter la mémoire. Par la suite, la fenêtre de données immédiatement suivante peut à son tour être chargée en mémoire centrale. Ceci implique un gain d'espace en mémoire par rapport à la stratégie CG, notamment avantageux pour le traitement des grands ensembles de données. La taille de la fenêtre peut être choisie en fonction des restrictions imposées par les limites des mémoires physiques. Dans les stratégies de classification du type CL, une fenêtre peut revenir en mémoire centrale si nécessaire, mais une seule fenêtre est en mémoire à la fois.

Supposons l'ensemble de données E partitionné en B paquets (fenêtres) $H^{(b)}$ non vides tels que $\forall i \neq j : H^{(i)} \cap H^{(j)} = \emptyset$ et $\bigcup_{b=1}^{B} H^{(b)} = E$, le processus d'exécution des stratégies CG , CL_2, et CL_3 est illustré sur la figure 3.6. Pour la classification globale CG, plusieurs itérations sur tout l'ensemble de données E sont effectuées jusqu'à la convergence de la méthode de classification. Pour la classification locale précédente CL_2, une seule itération est effectuée sur chaque fenêtre à la fois. Après le traitement de la dernière fenêtre, le processus reprend sur la première fenêtre jusqu'à la convergence de la méthode de classification. Cette pratique requiert un temps de transferts non négligeable des fenêtres entre le disque et la mémoire. Afin de minimiser le nombre d'allers-retours des fenêtres en mémoire, la classification locale dépendante CL_3 effectue plusieurs itérations sur une même fenêtre avant de passer à la fenêtre suivante. On peut ainsi espérer que la convergence de la méthode de classification sera atteinte plus rapidement.

Dans un cadre d'application des stratégies de classification sur un flux de données, quelques considérations se font nécessaires. Dans un flux de données (par exemple les données issues des fichiers log Web, des télécommunications, des capteurs, des indices boursiers, de la météorologie, du trafic routier ou des réseaux d'ordinateurs, etc.), les données arrivent *en ligne* avec un débit important. Cette masse trop importante de données est non bornée et ne peut pas être intégralement stockée ni en mémoire centrale ni en disque. Pour cette raison, les trois stratégies de classification illustrées sur la figure 3.6 sont impraticables sur un flux de données car elles supposent les fenêtres de données toujours présentes sur le disque.

La classification locale indépendante CL_1 reste ainsi la seule applicable dans le cadre de classification d'un flux de données puisque dabs cette stratégie, une fenêtre de données est placée une seule fois en mémoire.

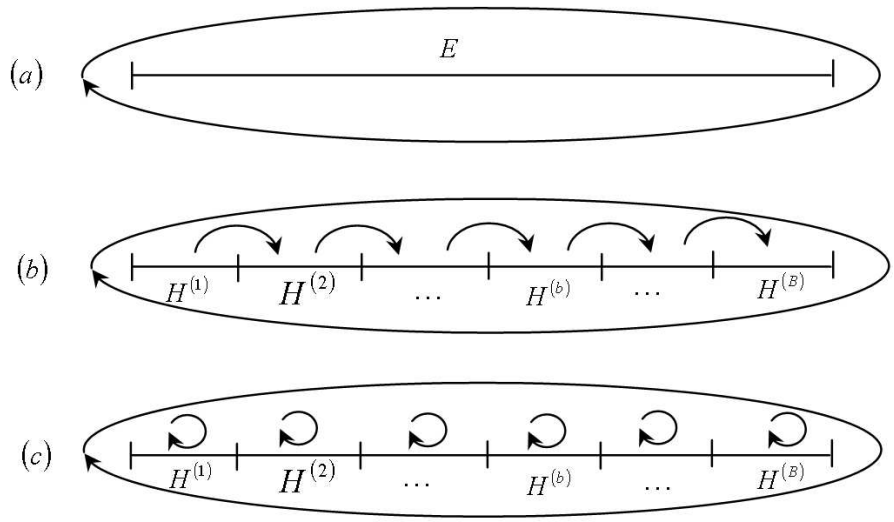

FIGURE 3.6 – Schéma d'application des stratégies : (a) Classification globale (CG) sur l'ensemble E, (b) Classification locale précédente CL_2 sur les paquets $H^{(b)}$ de E, (c) Classification locale dépendante CL_3 sur les paquets $H^{(b)}$ de E.

4.3 Application des stratégies de classification avec une méthode de classification

Dans cette section, nous décrivons la mise en place des stratégies de classification proposées dans la section précédente dans un cadre d'application avec la Méthode des Nuées Dynamiques (MND de [40] [42]), introduite dans la section 3.1 du chapitre 2.

Remarque : Pour l'application des stratégies de classification dites *locales* (CL_1,CL_2,CL_3), l'ensemble de données E doit être d'abord partitionné en B paquets (fenêtres) non vides $H^{(1)}, ..., H^{(B)}$, tels que $\forall i \neq j : H^{(i)} \cap H^{(j)} = \emptyset$ et $\bigcup_{b=1}^{B} H^{(b)} = E$.

Classification globale

Il s'agit du cadre classique d'application de la MND, initialisée avec des prototypes aléatoires, sur un ensemble de données E. Pour l'application de cette stratégie, tout le tableau de données associé à E doit être chargé en mémoire centrale.

Algorithme base de la MND

L'algorithme base de la MND se décline de la façon suivante :

On part d'un ensemble de prototypes $L^{(0)} = (L_1^0, \ldots, L_K^0) \in \mathcal{L}_K$.

On définit deux suites $v_t = \left(L^{(t)}, P^{(t)}\right)$ et $u_t = W\left(v_t\right)$, avec $P^{(t)} = f\left(L^{(t-1)}\right)$ et $L^{(t)} = g\left(P^{(t)}\right)$.

L'application $f : \mathcal{L}_K \to \mathcal{P}_K(E)$ est définie par :
$P^{(t)} = f\left(L^{(t-1)}\right)$ où $P^{(t)} = (C_1^{(t)}, ..., C_K^{(t)})$ et $C_k^{(t)} = \{i \in E / d\left(x_i, g_l^{(t-1)}\right) \leq d\left(x_i, g_k^{(t-1)}\right) \forall l, k \in \{1, ..., K\} : l \neq k\}$.
En cas d'égalité, i est affecté à la classe de plus petit indice.

L'application $g : \mathcal{P}_K(E) \to \mathcal{L}_K$ est définie par :
$L^{(t)} = g\left(P^{(t)}\right)$ avec

$$W\left(L^{(t)}, P^{(t)}\right) = \min_{L \in \mathcal{L}_K} W\left(L, P^{(t)}\right) \tag{3.4}$$

Remarque : Si $W(L, P) = \sum_{k=(1,\cdots,K)} S(L_k, C_k)$ avec $S(L_k, C_k) = \sum_{i \in C_k} w_i d^2(x_i, L_k)$, alors résoudre l'équation 3.4 est équivalent à résoudre pour chaque $k = 1, ..., K$:
$S\left(L_k, C_k\right) = \min_{L \in \mathcal{L}} S\left(L, C_k\right)$

Théorème 1 [40]

La suite u_t est décroissante et converge en un nombre fini d'interactions.
La suite v_t est convergente.

Démonstration de ce théorème dans le cas de $\mathcal{L} \equiv \Re^p$

Soit $v_{t-1} = \left(L^{(t-1)}, P^{(t-1)}\right)$ où
$L^{(t-1)} = \left(g_1^{(t-1)}, \ldots, g_K^{(t-1)}\right)$ avec $g_k^{(t-1)} \in \Re^p$ et $P^{(t-1)} = \left(C_1^{(t-1)}, \ldots, C_K^{(t-1)}\right)$.

Nous avons :
$W\left(L^{(t-1)}, P^{(t-1)}\right) = \sum_{k=1}^{K} S(g_k^{(t-1)}, C_k^{(t-1)}) = \sum_{k=1}^{K} \sum_{i \in C_k^{(t-1)}} w_i d^2\left(x_i, g_k^{(t-1)}\right)$

Soit $P^{(t)} = f(L^{(t-1)})$ où $P^{(t)} = \left(C_1^{(t)}, \ldots, C_K^{(t)}\right)$, alors nous avons :

$$W\left(L^{(t-1)}, P^{(t-1)}\right) = \sum_{k=1}^{K} \sum_{k'=1}^{K} \sum_{i \in \left(C_k^{(t-1)} \cap C_{k'}^{(t)}\right)} w_i d^2\left(x_i, g_k^{(t-1)}\right)$$

et

$$W\left(L^{(t-1)}, P^{(t)}\right) = \sum_{k'=1}^{K} \sum_{k=1}^{K} \sum_{i \in \left(C_{k'}^{(t)} \cap C_k^{(t-1)}\right)} w_i d^2\left(x_i, g_{k'}^{(t-1)}\right)$$

- Si i ne change pas de classe, nous avons $k = k'$ et $d\left(x_i, g_{k'}^{(t-1)}\right) = d\left(x_i, g_k^{(t-1)}\right)$.

- Si i change de classe, nous avons $k \neq k'$ et $d\left(x_i, g_{k'}^{(t-1)}\right) < d\left(x_i, g_k^{(t-1)}\right)$.

D'où si $P^{(t)} \neq P^{(t-1)}$, nous avons

$$W\left(L^{(t-1)}, P^{(t)}\right) \leq W\left(L^{(t-1)}, P^{(t-1)}\right) \tag{3.5}$$

Comme par construction de $L^{(t)} = g(P^{(t)})$, nous avons :

$$W\left(L^{(t)}, P^{(t)}\right) = \min_{L \in (\Re^p)^K} W\left(L, P^{(t)}\right)$$

D'où

$$W(L^{(t)}, P^{(t)}) \leq W(L^{(t-1)}, P^{(t)}) \tag{3.6}$$

Dans notre cas, d est la distance Euclidienne, alors $W(L,P) = \sum_{k=1}^{K} \sum_{i \in C_k} w_i ||x_i - g||^2$

$W(L,P) = \sum_{k=1}^{K} S(g, C_k)$ où $S(g, C_k) = \sum_{i \in C_k} w_i ||x_i - g||^2$.

La solution g_k telle que $S(g_k, C_k) = \min_{g \in \Re^p} \sum_{i \in C_k} w_i ||x_i - g||^2$ est la moyenne pondérée, d'où $g_k = \sum_{i \in C_k} \frac{w_i x_i}{\left(\sum_{i \in C_k} w_i\right)}$.

D'aprés les équations 3.5 et 3.6, nous avons :
$W(L^{(t)}, P^{(t)}) \leq W(L^{(t-1)}, P^{(t)}) \leq W\left(L^{(t-1)}, P^{(t-1)}\right)$.

La suite u_t est décroissante et comme $u_t >= 0$, alors elle est convergente. Si à l'étape t la suite u_t n'a pas convergé, alors toutes les partitions $P^{(1)}, \ldots, P^{(t)}$ sont différentes. Comme l'ensemble des partitions en K classes non vides est fini, alors le nombre d'étapes t est aussi fini.

Classification locale indépendante

Comme précisé en début de section, pour l'application des stratégies de classification locales, l'ensemble de données E est partitionné en B paquets (fenêtres) de données. Dans l'application de la stratégie de classification locale indépendante, l'ensemble de données traité par la MND, initialisée avec des prototypes aléatoires, se réduit à l'ensemble de données contenues dans chaque paquet $H^{(b)}$.

Classification locale "précédente"

Soit l'ensemble de données E de taille n partitionné en B paquets $H^{(b)}$ non vides. Les individus du paquet $H^{(b)}$ où $1 < b \leq B$ doivent être affectés aux prototypes issus du paquet immédiatement précédent $H^{(b-1)}$.

Dans cette approche, il est nécessaire de calculer les nouveaux prototypes qui seront envoyés au paquets suivant $H^{(b+1)}$. Pour ce faire, si l'on applique une seule étape de mise à jour des prototypes, cette stratégie correspond à celle utilisée par la Méthode des Nuées Dynamiques Séquentialisée (MNDS) [41], expliquée dans la section suivante.

Classification locale dépendante

Si on applique la MND avec la stratégie de *classification locale dépendante* sur chaque paquet (fenêtre) $H^{(b)}$, cette stratégie pourrait donner lieu à une nouvelle méthode de classification appelée : Méthode des Nuées Dynamiques Séquentialisée Optimisée (MNDSO).

La MNDSO est inspirée de la Méthode des Nuées Dynamiques Séquentialisée (MNDS proposée par [41]), mais diffère de cette dernière par le fait que la MNDSO optimise localement sur le paquet $H^{(b)}$ le critère $W\left(L^{(t-1)}, H^{(b)}, P^{(t-1)}\right)$ avant de passer au paquet suivant $H^{(b+1)}$, alors que la MNDS effectue une seule étape de mise à jour des prototypes après l'étape d'affectation des individus du paquet $H^{(b)}$ pour ensuite traiter les indivudus du paquet $H^{(b+1)}$.

La MNDS présente un grand avantage par rapport à la MND en ce qui concerne la quantité de mémoire requise par le processus de classification. Ceci rend possible le traitement des bases de données de taille importante pour lesquelles le chargement intégral en mémoire est impraticable.

L'idée de base de la MNDS est de repartir l'ensemble de données E en B paquets $H^{(1)}, ..., H^{(B)}$ tels que $\forall i \neq j : H^{(i)} \cap H^{(j)} = \emptyset$ et $\bigcup_{b=1}^{B} H^{(b)} = E$, et actualiser les centres des classes dès que tous les individus d'un paquet sont affectés, puis répéter ce processus sur le paquet suivant. Après l'affectation des individus du dernier paquet $H^{(B)}$ et de la mise à jour des centres des classes, le processus reprend en boucle sur le premier paquet $H^{(1)}$ jusqu'à la convergence du critère W. La MNDS suppose que tous les paquets de données sont toujours disponibles sur le disque et peuvent revenir en mémoire dès que nécessaire.

Convergence de la MNDS

On note $H^{(1)}, \ldots, H^{(B)}$ une partition en B classes non vides de E.
Soit $S(g, C_k) = \sum_{i \in C_k} w_i d^2(x_i, g_k)$ avec $g \in \Re^p$.
Le critère W s'écrit :
$W\left(L^{(t-1)}, E, P^{(t-1)}\right) = \sum_{b=1}^{B} \sum_{k=1}^{K} \sum_{k'=1}^{K} S\left(g_k^{(t-1)}, C_{k'}^{(t)} \cap C_k^{(t-1)} \cap H^{(b)}\right)$
Pour b fixé, nous avons :

$$\begin{aligned} W\left(L^{(t-1)}, E, P^{(t-1)}\right) &= \sum_{b'=1, b' \neq b}^{B} \sum_{k=1}^{K} S\left(g_k^{(t-1)}, C_k^{(t-1)} \cap H^{(b')}\right) + \\ &+ \sum_{k=1}^{K} \sum_{k'=1}^{K} S\left(g_k^{(t-1)}, C_{k'}^{(t)} \cap C_k^{(t-1)} \cap H^{(b)}\right) \end{aligned}$$

Construction de $P^{(t)}$

L'individu $i \in (C_k^{(t-1)} \cap H^{(b)})$ sera affecté à la classe k' telle que $k' = arg \min_{l=1,\ldots,K} d(x_i, g_l^{(t-1)})$.

- Si i ne change pas de classe, nous avons $k = k'$ d'où $d\left(x_i, g_{k'}^{(t-1)}\right) = d\left(x_i, g_k^{(t-1)}\right)$.

- Si i change de classe, nous avons $k \neq k'$ d'où $d\left(x_i, g_{k'}^{(t-1)}\right) < d\left(x_i, g_k^{(t-1)}\right)$.

D'où :

$$\begin{aligned}
\sum_{k'=1}^{K} \sum_{i \in \left(C_{k'}^{(t)} \cap H^{(b)}\right)} w_i d^2(x_i, g_{k'}^{(t-1)}) &\leq \sum_{k=1}^{K} \sum_{i \in \left(C_{k}^{(t-1)} \cap H^{(b)}\right)} w_i d^2(x_i, g_{k}^{(t-1)}) \\
\sum_{k'=1}^{K} S\left(g_{k'}^{(t-1)}, C_{k'}^{(t)} \cap H^{(b)}\right) &\leq \sum_{k=1}^{K} S\left(g_{k}^{(t-1)}, C_{k}^{(t-1)} \cap H^{(b)}\right) \\
W\left(L^{(t-1)}, E, P^{(t)}\right) &\leq W\left(L^{(t-1)}, E, P^{(t-1)}\right) \qquad (3.7)
\end{aligned}$$

Construction de $L^{(t)}$

$W\left(L^{(t)}, E, P^{(t)}\right) = \min_{L \in (\Re^p)^K} W\left(L, E, P^{(t)}\right)$ où $L^{(t)} = \left(g_1^{(t)}, ..., g_l^{(t)}, ..., g_K^{(t)}\right)$

Comme le critère global est additif en fonction des paquets, il suffit de calculer les moyennes globales en fonction des modifications des moyennes locales par les formules [41] :

$$\begin{aligned}
\mu_k^{(t)} &= \mu_k^{(t-1)} + \epsilon_i w_i \quad \text{avec} \quad \mu_k^{(0)} = \sum_{i \in C_k^0} w_i \\
g_k^{(t)} &= \frac{\mu_k^{(t-1)} g_k^{(t-1)} + \sum_{i \in H^{(b)}} \epsilon_i w_i x_i}{\mu_k^{(t)}} \text{ où} \\
\epsilon_i &= +1 \quad \text{si} \quad i \in C_k^{(t)} \quad \text{et} \quad i \notin C_k^{(t-1)} \\
\epsilon_i &= -1 \quad \text{si} \quad i \notin C_k^{(t)} \quad \text{et} \quad i \in C_k^{(t-1)} \\
\epsilon_i &= 0 \quad \text{si} \quad i \in C_k^{(t)} \quad \text{et} \quad i \in C_k^{(t-1)}
\end{aligned}$$

D'où :

$$W\left(L^{(t)}, E, P^{(t)}\right) \leq W\left(L^{(t-1)}, E, P^{(t)}\right) \qquad (3.8)$$

A partir des équations 3.7 et 3.8, nous avons :

$$\begin{aligned}
W\left(L^{(t)}, E, P^{(t)}\right) &\leq W\left(L^{(t-1)}, E, P^{(t-1)}\right) \\
u_t &\leq u_{t-1}
\end{aligned}$$

En effet, la MNDS suppose que tous les paquets de données soient stockés sur le disque et puissent être consultés à plusieurs reprises. Cette méthode doit effectuer plusieurs passages sur un même paquet jusqu'à sa convergence. La MNDSO propose de réduire le nombre de mises en mémoire de chaque paquet (fenêtre) de données. Comme mentionnée précédemment, l'idée de la MNDSO est d'optimiser localement sur le paquet $H^{(b)}$ le critère $W\left(L^{(t-1)}, H^{(b)}, P^{(t-1)}\right)$ avant de traiter les données du paquet $H^{(b+1)}$.

Algorithme base de la MNDSO

On vérifie si $W\left(L^{(t)}, H^{(b)}, P^{(t)}\right) < W\left(L^{(t-1)}, H^{(b)}, P^{(t-1)}\right)$. Si cela est vrai, on recommence les deux étapes d'*affectation* et *représentation* sur le paquet $H^{(b)}$. Si $W\left(L^{(t)}, H^{(b)}, P^{(t)}\right) = W\left(L^{(t-1)}, H^{(b)}, P^{(t-1)}\right)$, alors on passe au paquet $H^{(b+1)}$.

Théorème 2

Les trois algorithmes MND, MNDS et MNDSO font décroître le même critère W et convergent vers les mêmes solutions.

Démonstration

Dans ([41] page 36) se trouve la démonstration que la MND et la MNDS convergent vers les mêmes solutions.

Soit (L^*, P^*) une solution pour la MND. Dans ce cas, nous avons :
$\forall L \in \left(\Re^p\right)^K : W\left(L, P^*\right) \geq W\left(L^*, P^*\right)$ et
$\forall P \in \mathcal{P}_K(E) : W\left(L^*, P\right) \geq W\left(L^*, P^*\right)$

Comme $W\left(L, P^*\right) \geq W\left(L^*, P^*\right)$, nous avons pour le paquet $H^{(b)}$:
$W\left(L, H^{(b)}, P^*\right) \geq W\left(L^*, H^{(b)}, P^*\right)$
C'est-à-dire qu'il n'existe pas de $i \in \left(H^{(b)} \cap C_k^*\right)$ tel que $d\left(x_i, g_{k'}\right) < d\left(x_i, g_k\right)$ pour $k^{'} \neq k$. Il est donc impossible qu'un individu i change de classe.

De même, si (L^{**}, P^{**}) est solution de la MNDSO, cela implique que les partitions locales de $H^{(b)}$ restent inchangeables par rapport au prototype L^{**} et la valeur $\left(L^{**}, H^{(b)}, P^{**}\right)$ est minimale. Ainsi, le critère $W\left(L^{**}, P^{**}\right)$ est aussi

minimal.

5 Approche de classification pour la détection et le suivi des changements sur des données évolutives

Les analyses préliminaires détaillées dans la section 4 du présent chapitre constituent des esquisses initiales fournissant d'importantes pistes pour la définition d'une stratégie efficace pour la détection et le suivi des changements sur des données évolutives. A l'issue des analyses des résultats obtenus lors des expérimentations (cf. chapitre 5), nous avons pu formuler une approche de détection et de suivi des changements sur des données évolutives. La description de cette approche fait l'objet de la présente section.

5.1 Présentation de l'approche

Soit W^{t-1} une fenêtre et W^t la fenêtre non recouvrante qui la suit dans le temps. Pour démarrer la procédure, une classification non supervisée, initialisée avec des prototypes aléatoires, doit être appliquée sur les données contenues à l'intérieur de la fenêtre W^{t-1}. Soit $G^{t-1} = (g_1, \ldots, g_c, \ldots, g_{k^{t-1}})$ l'ensemble des représentants (prototypes) finaux des clusters issus de cette classification.

Au pas suivant, notre approche consiste à affecter les individus de la fenêtre suivante W^t aux prototypes de G^{t-1}, ceci afin d'obtenir une première partition P_1^t contenant k^{t-1} clusters sur les individus de la fenêtre W^t. Cette partition correspond à celle obtenue localement par la stratégie de *classification locale précédente* (cf. section 4.1).

Par la suite, une méthode de classification non supervisée, initialisée avec des prototypes aléatoires, doit être appliquée sur les individus de la fenêtre W^t. Ce qui fournit, à la convergence de la méthode de classification, une deuxième partition P_2^t contenant k^t clusters sur les individus de la fenêtre W^t. Cette partition correspond à celle obtenue localement par la stratégie de *classification locale indépendante* (cf. section 4.1). Le schéma de notre approche de classification pour la détection des changements est illustré par la figure 3.7.

Il est importante de remarquer que la partition P_1^t porte le reflet de la structure

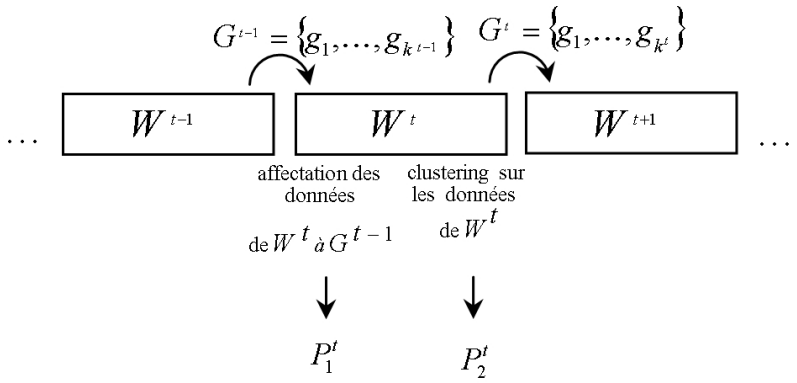

Figure 3.7 – Déroulement de l'approche de classification pour la détection et le suivi des changements.

classificatoire de la fenêtre précédente W^{t-1} sur les données de la fenêtre courante W^t, cette partition reflète ainsi l'influence du passé sur les données plus récentes dans le temps. Alors que la partition P_2^t ne reçoit aucune influence du passé et reflète l'organisation actuelle des données dans la fenêtre courante.

Les changements qui peuvent avoir lieu entre deux fenêtres voisines seront repérés par la comparaison des deux partitions P_1^t et P_2^t à l'aide des critères de comparaison des partitions (présentés dans la section 5 du chapitre 2). Plus petite est la valeur obtenue par ces critères, plus importante est la différence entre les deux partitions comparées. Notons que la détection des changements est effectuée à partir de la comparaison des partitions issues d'un même ensemble d'individus contenus à l'intérieur de la fenêtre courante W^t. La détection des ruptures entre deux périodes consécutives de temps reposera ainsi sur la confrontation des deux partitions nées d'un même ensemble de données différemment segmenté.

Parmi les trois stratégies de classification locale proposées dans la section 4 du présent chapitre, la *classification locale indépendante* CL_1 est la plus couteuse en temps de calcul, ceci dû à plusieurs initialisations avec des prototypes aléatoires et à l'exécution de la méthode de classification jusqu'à sa convergence sur chaque fenêtre de données. Malgré cela, la stratégie de classification locale indépendante se montre la plus adaptée au problème de détection de changements entre deux fenêtres consécutives, ce qui justifie notre choix pour cette stratégie de classification dans notre approche de détection de changements. Ce fait est constaté par les résultats des expérimentations décrites dans la section 3.1 du chapitre 5. En effet, la classification locale indépendante est capable de détecter des changements dans le temps qui

ne sont pas repérés par les autres stratégies de classification investiguées. Dans cette stratégie, les prototypes initiaux sont indépendants des prototypes issus des classifications sur les fenêtres précédentes, la classification locale obtenue est ainsi fidèle aux données dans la fenêtre courante.

Dans le cas d'application avec les Nuées Dynamiques, les méthodes MND, MNDS et MNDSO (cf. section 4.3 du présent chapitre) supposent que la population analysée suit une loi P constante. Elles réalisent une mise à jour des prototypes globaux des classes et convergent vers une même solution. De l'autre côté, la stratégie de classification *locale indépendante* réalise une mise à jour des prototypes locaux des classes sur le paquet (fenêtre) de données analysé. Elle permet ainsi de faire ressortir les possibles évolutions de la loi P d'un paquet de données à l'autre.

L'un des points forts de l'approche proposée concerne son indépendance de la méthode de classification. Il est cependant judicieux d'utiliser la même méthode de classification sur toutes les fenêtres une fois que différentes méthodes de classification peuvent aboutir à différentes partitions sur un même ensemble de données, ceci en fonction d'artefacts particuliers à chaque méthode. La seule exigence concernant la méthode de classification à être utilisée dans cette approche repose sur le fait que celle-ci doit fournir un représentant (prototype) pour chaque cluster identifié. Notre approche est ainsi capable - grâce à son traitement de données par paquets (fenêtres) - de rendre incrémentale n'importe quelle méthode de classification non supervisée respectant cette contrainte.

5.2 Interprétation sémantique des changements

Dans le cadre de cette thèse, nous ne nous intéressons pas uniquement à la détection des changements sur des données évolutives, mais aussi et surtout à l'interprétation des changements repérés. En d'autres termes, nous cherchons à comprendre ce qui se passe dans le système afin d'identifier les possibles causes des ruptures et ainsi pouvoir donner des éléments de réponse à la problématique soulevée.

Dans un processus de classification traditionnelle, après l'application de la méthode de classification sur les données d'usage et une fois toutes les navigations classées dans des clusters, plusieurs informations peuvent en être extraites, à savoir :

– chaque cluster rassemble des navigations appartenant à des individus ayant visité de pages similaires et qui partagent ainsi les mêmes préférences ;

- un profil d'usage peut être défini pour chaque cluster, soit par son prototype soit par l'élément le plus représentative du groupe ;
- les clusters ayant un effectif important correspondent aux profils d'usage les plus populaires ;
- les clusters ayant un faible effectif correspondent aux profils d'usage minoritaires.

Ces informations sont très intuitives et peuvent être facilement extraites à partir d'une simple analyse sur la partition de données. Or, dans le cadre d'analyse des données évolutives, de telles informations s'avèrent insuffisantes en raison de la nature dynamique intrinsèque à ce type de données. Dans ce contexte, la prise en compte de l'évolution subie par les données est indispensable afin de pouvoir répondre à de questions du type : *'A quelle période précise de temps un nouveau cluster de comportement émerge ?'*, *'A quelle période de temps un cluster de comportement disparait ?'*, *'Combien de clusters de comportement survivent d'une période de temps à l'autre ?'*, *'Parmi ceux qui survivent, lesquels subissent des divisions (scissions) ou des fusions (absorptions) ?'*

Dans un environnement dynamique évolutif, les types de changements subis par les clusters au cours du temps peuvent être discriminés comme suit :

- **Disparition d'un cluster** : un cluster existant dans la partition P_1^t n'est plus présent dans la partition P_2^t ;
- **Apparition d'un cluster** : un cluster inexistant dans la partition P_1^t émerge dans la partition P_2^t ;
- **Scission ou division d'un cluster** : un certain nombre d'individus appartenant à un cluster de la partition P_1^t se retrouve dans d'autres clusters de la partition P_2^t. Dans le domaine de l'usage du Web, ceci peut indiquer un changement de préférence de pages d'un certain groupe d'utilisateurs ;
- **Fusion ou absorption de deux ou plusieurs clusters** : les individus de différents clusters de la partition P_1^t migrent vers un même cluster de la partition P_2^t. Dans le domaine du *marketing*, cet événement pourrait indiquer quand des clients de marques distinctes partagent les mêmes préférences d'achat ;
- **Pas de changement externe** : dans ce dernier cas, le cluster est dit survivant dans la fenêtre de temps suivante. Il peut cependant subir des changements internes liés par exemple à son effectif (rétrécissement ou grossissement). Cet événement pourrait indiquer le changement de popularité de certains produits de commerce, par exemple.

Afin d'interpréter le type de changement subi par un cluster et d'en réaliser un suivi au cours du temps, notre approche implémente des fonctionnalités additionnelles capables de repérer les transformations subies par un cluster de comportement au cours du temps. L'idée est de s'appuyer sur la comparaison des clusters par l'intermédiaire de leur *extension*, c'est-à-dire, l'ensemble d'individus affectés aux clusters, en opposition aux techniques de comparaison basées sur l'*intension*, où la comparaison est effectuée sur les prototypes des clusters en question.

Le principe de la discrimination des changements subis par un cluster est décrit comme suit. Soit $P_1^t = \{C_{11}^t, \cdots, C_{1i}^t, \cdots, C_{1K^{t-1}}^t\}$ la partition obtenue par affectation des individus dans la fenêtre courante W^t aux prototypes en provenance de la fenêtre W^{t-1}. Cette partition a ainsi un total de K^{t-1} clusters. Soit $P_2^t = \{C_{21}^t, \cdots, C_{2j}^t, \cdots, C_{2K^t}^t\}$ la partition obtenue après l'application de la méthode de classification sur les individus de la fenêtre courante W^t. La partition P_2^t a un total de K^t clusters. Notre objectif est de comparer les partitions P_1^t et P_2^t obtenues à partir d'un même ensemble d'individus, et ce afin de repérer la correspondance entre les clusters de ces deux partitions. La correspondance des clusters se fait à l'aide du *rappel* (cf. équation 2.20 du chapitre 2) qui représente le pourcentage d'effectifs conservés entre un cluster C_{1i}^t de la partition P_1^t et un cluster C_{2j}^t de la partition P_2^t.

Soit $survivalThreshold$ un seuil indiquant le pourcentage minimum d'effectifs à être conservés par un cluster en cas de survie et $splitThreshold$ un seuil indiquant le pourcentage minimum d'effectifs à être conservés par un cluster originaire d'une scission. Ce dernier seuil est utilisé pour éviter le cas dégénératif des clusters très fragmentés. Ces deux seuils doivent être fixés en tant que paramètre d'entrée dans notre approche.

- **Définition 1.** Un cluster de P_1^t est dit **survivant** dans la partition P_2^t quand il conserve un minimum d'effectifs, c'est-à-dire, $rappel(C_{1i}^t, C_{2j}^t) >= survivalThreshold$.
- **Définition 2.** Le cluster C_{2j}^t de P_2^t correspondant à un cluster C_{1i}^t de P_1^t est celui qui conserve le plus grand effectif, c'est-à-dire, $arg \max_{j \in (1,\cdots,K^t)} rappel(C_{1i}^t, C_{2j}^t)$ avec $rappel(C_{1i}^t, C_{2j}^t) >= survivalThreshold$.
- **Définition 3.** Un cluster de P_1^t est dit **disparu** quand il n'a pas de cluster correspondant dans P_2^t.
- **Définition 4.** Un cluster de la partition P_1^t est dit **scissionné** quand un minimum de ces effectifs migre vers deux ou plusieurs clusters de la partition

P_2^t, c'est-à-dire, $rappel(C_{1i}^t, C_{2j}^t) >= splitThreshold$.

- **Définition 5.** Un cluster de la partition P_1^t est dit cluster **fusionné** ou **absorbé** quand il est un cluster survivant dont le cluster correspondant dans la partition P_2^t est également cluster correspondant d'au moins un autre cluster de P_1^t.
- **Définition 6.** Un cluster de P_2^t est dit **nouveau** ou **apparu** quand il n'est cluster correspondant d'aucun cluster de P_1^t.

Durant nos expérimentations, nous avons fixé $survivalThreshold = 0.5$ et $splitThreshold = 0.4$, ceci implique : (i) un cluster survivant doit conserver au minimum 50% de son effectif dans la fenêtre de temps suivante et (ii) les clusters issus d'une scission doivent conserver au minimum 40% de l'effectif du cluster père.

Deux cas particuliers sont à considérer lors de la discrimination d'un cluster en tant que *survivant* ou *scissionné*, à savoir :

1. Si un cluster de P_1^t supposé *survivant* présente au moins un autre cluster de P_2^t contenant une importante partie de ces effectifs, alors le cluster en question est marqué *scissionné* à la place de survivant. Pour mieux comprendre ce cas de figure, considérons l'exemple de la figure 3.8, où les individus d'un cluster A migrent vers deux clusters distincts. Soit $survivalThreshold = 0.5$ et $splitThreshold = 0.4$. Dans un premier temps, l'on pourrait commettre l'erreur de marquer le cluster A' en tant que cluster correspondant au cluster originel A, vu que ce premier satisfait la condition de survie $rappel(A, A^{'}) >= survivalThreshold$. Or, le cluster A" contient un effectif non négligeable satisfaisant la condition de scission $rappel(A, A^{''}) >= splitThreshold$. Dans notre approche, la démarche adoptée est de marquer le cluster de départ comme scissionné et signaler les clusters résultants de cette scission. Ce cas est traité par la condition présente à la ligne 35 de l'algorithme d'interprétation sémantique.
2. Si pour un cluster de P_1^t, il existe plus d'un cluster de P_2^t satisfaisant la condition $rappel(C_{1i}^t, C_{2j}^t) >= survivalThreshold$, il sera marqué en tant que cluster scissionné. Ce cas de figure est traité par la condition présente à la ligne 34 de l'algorithme d'interprétation sémantique.

L'algorithme d'interprétation sémantique mis en place par notre approche se déroule comme suit :

```
1 totalSurvivals ← 0   // nombre de clusters survivants
2 totalSplits ← 0   // nombre de clusters scissionnés
```

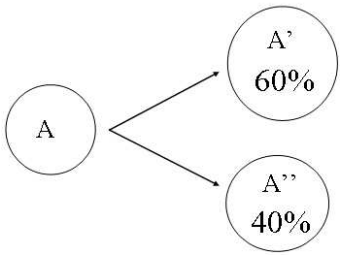

Figure 3.8 – Exemple d'un cluster A scissionné dans deux clusters A' et A''. Ceux-ci contiennent respectivement 60% et 40% des effectifs originels.

```
totalAbsorptions ← 0  // nombre de clusters fusionnés ou absorbés
totalDisapperances ← 0 // nombre de clusters disparus
totalApperances ← 0 // nombre de clusters apparus
Pour i dans l'intervalle[1, K^{t-1}] {
   maxRecall ← 0
   correspondentCluster[i] ← −1
   totalSplitCandidates ← 0
   totalSurvivalCandidates ← 0
   Pour j dans l'intervalle[1, K^t] {
      candidatesMatrix[i, j] ← 0
      recall ← recall(i, j)
      Si (recall >= thresholdSurvival) {
         Si (recall > maxRecall) {
            maxRecall ← recall
            correspondentCluster[i] ← j
            candidatesMatrix[i, j] ← 1 // survival code
            totalSurvivalCandidates ← totalSurvivalCandidates + 1
         }
      } Sinon si (recall >= thresholdSplit) {
         candidatesMatrix[i, j] ← 2 // split code
         totalSplitCandidates ← totalSplitCandidates + 1
      }
   }
   splitList ← ∅
   Si (totalSurvivalCandidates = 0) et (totalSplitCandidates <= 1) {
      // the cluster disappeared
      totalDisapperances ← totalDisapperances + 1
      correspondentCluster[i] ← −1 // disappearance code
      si (totalSplitCandidates = 1) {
            onlyOneSplitCandidate[i] ← true
      }
```

```
34    } Sinon si ((totalSurvivalCandidates > 1) ou
35                (totalSurvivalCandidates = 1) et (totalSplitCandidates > 0) ou
36                (totalSplitCandidates > 1)) {
37                // the cluster split
38                totalSplits ← totalSplits + 1
39                correspondentCluster[i] ← −2 // split code
40                 Si (totalSurvivalCandidates >= 1) {
41                      Pour j dans l'intervalle[1, K^t] {
42                         Si (candidatesMatrix[i, j] = 1) {
43                             // cluster was indeed split
44                             candidatesMatrix[i, j] ← 2
45                         }
46                      }
47                 }
48                 Pour j dans l'intervalle[1, K^t] {
49                         Si (candidatesMatrix[i, j] = 2) {
50                            splitList ← splitList ∪ j
51                         }
52                 }
53      }
54  }
55  newClusters ← ∅
56  Pour j dans l'intervalle[1, K^t] {
57    absorptionList ← ∅
58    comesFromSplit ← false
59    Pour i dans l'intervalle[1, K^(t−1)] {
60      Si (candidatesMatrix[i, j] = 1) {
61        absorptionList ← absorptionList ∪ i
62      } Sinon Si (candidatesMatrix[i, j] = 2){
63        comesFromSplit ← true
64        Si (onlyOneSplitCandidate[i] = true) {
65          totalApperances ← totalApperances + 1
66          newClusters ← newClusters ∪ j
67          Sortir de la boucle la plus interne
68        }
69      }
70    }
71    Si (|absorptionList| = 0) et (comesFromSplit = false) { // new cluster
72      totalApperances ← totalApperances + 1
73      newClusters ← newClusters ∪ j
74    } Sinon Si (|absorptionList| = 1) { // the cluster survived
75                  totalSurvivals ← totalSurvivals + 1
```

```
76              } Sinon Si (|absorptionList| > 1) {
77                  // the clusters where absorbed
78                  totalAbsorptions ← totalAbsorptions + |absorptionList|
79              }
80 }
```

Dans cet algorithme, $candidatesMatrix$ est une matrice auxiliaire de codage contenant K^{t-1} lignes et K^t colonnes. La cellule (i, j) de cette matrice contient une valeur indicative du possible changement subi par le cluster i de la partition P_1^t vis-à-vis du cluster j de la partition P_2^t. Dans cette matrice, le code 1 indique un cluster candidat à la survie, le code 2 indique un cluster candidat à la scission.

La variable $correspondentCluster$ a pour but d'identifier le cluster j de la partition P_2^t correspondant au cluster i de la partition P_1^t. Si le cluster i a disparu dans la fenêtre suivante, le code -1 est utilisé, si le cluster i a subi une scission, alors le code -2 est utilisé et la liste des clusters issus de cette scission est enregistrée dans la variable $splitList$.

Le premier pas de l'algorithme consiste à chercher, pour chaque cluster de la partition P_1^t (contenant K^{t-1} clusters) le cluster correspondant dans la partition P_2^t (contenant K^t clusters). Ce rapprochement est fait en fonction du *rappel* entre tous les clusters de ces deux partitions (cf. section 5 du chapitre 2). Le cluster correspondant dans la fenêtre de temps suivante sera celui qui présente la plus grande valeur de rappel (lignes 15-20 de l'algorithme).

Si un cluster n'a pas de cluster correspondant dans la fenêtre de temps suivante où s'il possède un unique candidat issu d'une scission, alors le cluster est marqué *disparu* (condition présente à la ligne 27 de l'algorithme).

Une fois les clusters survivants retrouvés, le pas suivant consiste à vérifier s'il y a eu de clusters fusionnés ou de clusters émergeant dans la fenêtre de temps suivante. Cette vérification est faite sur les lignes 55-80 de l'algorithme. Un nouveau cluster peut naître soit des effectifs minoritaires en provenance des clusters scissionnés (dans ce cas, la condition $recall >= thresholdSplit$ n'a pas été satisfaite, ce qui est représenté par la condition $comesFromSplit = false$ à la ligne 71 de l'algorithme), soit d'un seul candidat à scission (ligne 64 de l'algorithme). Les clusters fusionnés sont détectés à partir de la vérification du nombre de clusters i de la partition P_1^t ayant comme correspondant un même cluster j dans la partition P_2^t (lignes 60-61 de l'algorithme). Si ce nombre est plus grand que 1, alors il y a eu une fusion de clusters (ligne 76 de l'algorithme), sinon le cluster en question est marqué comme

survivant (ligne 74 de l'algorithme).

Le *rappel* peut également être exprimé en termes de probabilité, comme montre l'équation 3.9. Cette équation exprime la probabilité conditionnelle d'occurrence du cluster C_{2j}^t de la partition P_2^t étant donné le cluster C_{1i}^t de la partition P_1^t.

$$rappel(C_{1i}^t, C_{2j}^t) = P(C_{2j}^t / C_{1i}^t) = \frac{P(C_{1i}^t \cap C_{2j}^t)}{P(C_{1i}^t)} \tag{3.9}$$

Considérons l'exemple de la figure 3.9 qui montre la migration des individus (navigations) entre fenêtres voisines. Pour ce jeu de données d'usage, nous appliquons une fenêtre temporelle de taille égale à 1 mois. Les traces d'usage sont enregistrés de juillet 2002 jusqu'à mai 2003, ce qui nous donne un total de 11 fenêtres temporelles. Les résultats obtenus sur ce jeu de données seront détaillés dans la section 3.1 du chapitre 5.

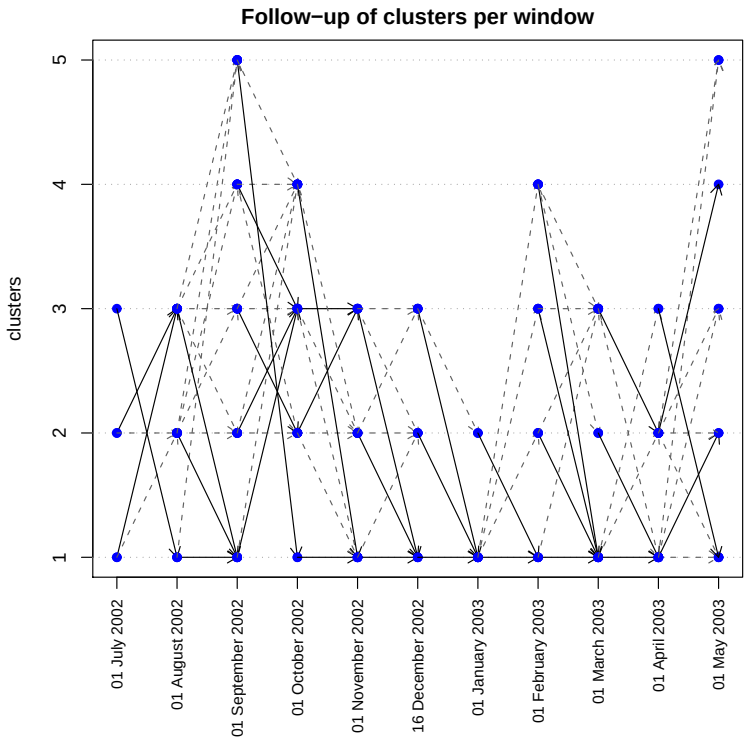

Figure 3.9 – Représentation graphique de la migration des individus des clusters de partitions issues des fenêtres voisines.

Sur la figure 3.9, l'axe des abscisses représente les fenêtres des données analysées.

Chaque fenêtre est labélisée par la date d'occurrence du premier clic dans la première navigation du mois en question. L'axe des ordonnées représente le label des clusters repérés. Le nombre total de clusters varie entre 2 et 5 sur chaque fenêtre.

Les traits continus représentent le trajet migratoire de la majorité des effectifs d'un cluster entre deux fenêtres voisines, c'est-à-dire, $arg \max_{j \in (1,\cdots,K^t)} P(C_{2j}^t / C_{1i}^t)$. Les traits pointillés représentent le trajet migratoire des autres effectifs du cluster en question, c'est-à-dire, $P(C_{2j}^t / C_{1i}^t) \neq 0$. Par exemple, les partitions issues des classifications sur la première fenêtre (juillet 2002) et la deuxième fenêtre (août 2002) analysées présentent toutes les deux 3 clusters. Le cluster 3 du mois de juillet 2002 a survécu et a comme correspondant le cluster 1 du mois août 2002. Les clusters 1 et 2 de juillet 2002 se sont fusionnés dans le cluster 3 du mois août 2002. Le cluster 2 du mois août 2002 est apparu en fonction de la migration des effectifs minoritaires des clusters 1 et 2 du mois de juillet 2002.

5.3 Caractérisation et avantages de l'approche

Les caractéristiques générales ainsi que les avantages de l'approche proposée pour la détection et le suivi des changements sur des données évolutives peuvent être discriminés comme suit :

- **Stratégie du type *diviser pour régner*** : l'une des stratégies les plus efficaces pour résoudre un problème de grande dimension consiste à le diviser en sous-parties (modules) et en suite s'attaquer à chacune de ses parties séparément. Notre approche se base sur ce principe en appliquant le fenêtrage sur les données à classifier. Les efforts sont ainsi concentrés sur les données contenues à l'intérieur de chaque fenêtre à la fois. Cette stratégie permet de réduire les coûts (mémoire physique, allocation du microprocesseur, etc.) liés au traitement de l'ensemble de données originel car nous concentrons l'analyse sur une partie des données disponibles.
- **Réduction de la complexité et du temps d'exécution de la méthode de classification** : l'une des conséquences directes de l'application de la méthode de classification sur des fenêtres de données concerne la réduction de la complexité et du temps d'exécution si la méthode de classification utilisée a une complexité non linéaire. Considérons une méthode de classification d'ordre de complexité égal à $O(n^2)$. Le fait de partitionner l'ensemble de données à traiter en B fenêtres (cf. figure 3.10) implique une réduction de la complexité

de la méthode, celle-ci sera d'ordre $BO((\frac{n}{B})^2)$. Le tableau 3.3 montre la réduction des complexités pour différents valeurs de B sur un ensemble de données de taille 10^6.

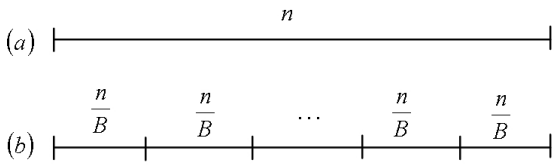

Figure 3.10 – (a) ensemble de données initial (b) ensemble de données partitionné en B fenêtres.

n	B (nombre de fenêtres)	$O(n^2)$	$BO((\frac{n}{B})^2)$
10^6	1	10^{12}	$1\text{x}10^{12} = 10^{12}$
10^6	10	10^{12}	$10\text{x}10^{10} = 10^{11}$
10^6	100	10^{12}	$100\text{x}10^{8} = 10^{10}$
10^6	1000	10^{12}	$1000\text{x}10^{3} = 10^{9}$

Table 3.3 – Valeurs des complexités pour différents valeurs de B sur un ensemble de données de taille 10^6.

- **Indépendance de la méthode de classification** : comme nous l'avons mentionné précédemment, l'approche proposée est totalement indépendant de la méthode de classification utilisée. Cette première se place sur un plus haut niveau d'abstraction et fait usage uniquement de la structure classificatoire et de l'ensemble de prototypes fournis par cette dernière.
- **Processus de détection des changements basé sur l'extension** : la stratégie adoptée pour la détection des changements dans notre approche repose sur la comparaison de deux partitions issues d'un même ensemble d'individus (contenus à l'intérieur d'une fenêtre). Cette comparaison est effectuée à l'aide d'un tableau de contingence construit à partir des deux partitions comparées. Pour ce faire, seule l'extension de la classification est utilisée. En choisissant les critères basés sur l'extension, la comparaison des partitions devient indépendante du type de représentation appliquée aux classes ainsi que des possibles imprécisions ou approximations engendrées par les critères de comparaison basés sur l'intension. La comparaison des partitions ne dépend que des individus eux-mêmes.
- **Résumé des données au cours du temps** : au fur et à mesure que l'approche se déroule et des fenêtres de données sont traitées, les résultats en-

gendrés par les analyses sont enregistrés dans une base de données relationnelle (cf. section 3 du chapitre 4), et ce de façon transparent et automatisée vis-à-vis de l'utilisateur final. Ainsi, un aperçu statique de la configuration classificatoire obtenue à chaque fenêtre de temps peut être obtenu par simple consultation de la base de données. Cette base de données enregistre également des informations concernant les paramètres de la méthode, à être définis par l'analyste selon les objectifs visés.

- **Application des algorithmes non-incrémentaux dans un processus incrémental** : l'approche proposée rend possible l'insertion, dans un processus incrémental, des algorithmes initialement conçus pour opérer sur tout l'ensemble des données disponibles. Ceci permet au système de détection de changement de bénéficier du potentiel des algorithmes non incrémentaux dans un processus additif.
- **Applicabilité dans un processus opérationnel** : L'approche proposée, tel qu'elle est conçue actuellement, peut être intégrée dans un processus opérationnel. Le système opère de façon incrémental et peut être arrêté/redémarré en toute indépendance. Les résultats étant enregistrés dans une base de données, le système est alimenté au fur et à mesure que des nouvelles données sont présentées au système. L'analyse reprend avec le mise en mémoire des prototypes issus de la classification sur la dernière fenêtre analysée et continue avec l'intégration des nouvelles fenêtres de données.
- **Généricité d'application dans d'autres domaines** : L'approche proposée, si bien qu'initialement conçue et motivée par le dynamisme lié aux données d'usage du Web, peut être extensible à d'autres domaines d'application. La seule restriction consiste au fait que les données doivent avoir un ordre d'arrivée. Quelques exemples d'analyse de données autres que les données du Web sont décrits dans le chapitre 5.

6 Synthèse

Au cours de ce chapitre, nous avons détaillé les contributions issues des travaux réalises dans le cadre de cette thèse. Ces contributions reposent sur la modélisation des données d'usage, la génération automatique de données évolutives d'usage, les analyses exploratoires des stratégies de classification dont le but a été de four-

nir des pistes et des éléments de réponse pour la problématique de détection de changements, et finalement, la proposition d'une approche de classification pour la détection et le suivi des changements sur des données évolutives.

Les expérimentations présentées dans le chapitre 5 décrivent les résultats issus des analyses exploratoires et permettent également de valider l'efficacité de l'approche proposée pour la détection et le suivi des changements sur des données évolutives.

Chapitre 4

Modélisation

1 Introduction

Avant de programmer une application, il est primordial d'organiser les idées concernées, les documenter et les modulariser en définissant des étapes. Cette démarche antérieure à l'écriture de l'application concerne la *modélisation* et son produit s'appelle un *modèle*. Modéliser un système avant sa réalisation permet de mieux comprendre le fonctionnement du système. C'est également un bon moyen de maîtriser sa complexité et d'assurer sa cohérence. Un modèle est une représentation abstraite et simplifiée d'une entité (phénomène, processus, système, etc.) du monde réel en vue de le décrire, de l'expliquer et de le communiquer à d'autres personnes. Les modèles peuvent ainsi faire l'objet de document de spécification ou de conception. Le modèle est enfin indispensable pour assurer un bon niveau de qualité et une maintenance efficace.

Ce chapitre a pour but de présenter la modélisation de notre approche telle quelle a été envisagée. La section 2 présente les diagrammes de packages et de classes construits à l'aide de l'UML. La section 3 présente le modèle conceptuel des données, et finalement, dans la section 4, nous spécifions les requêtes SQL utilisées pour l'extraction des variables statistiques descriptives des navigations (présentées dans la section 3 du chapitre 2) à partir d'un entrepôt contenant les données d'usage prétraitées.

2 Modélisation UML

Pour modéliser une application, il est nécessaire de choisir un langage commun, précis et connu par tous les membres d'une équipe. L'UML (Unified Modelling Language, en anglais, ou langage de modélisation unifié, en français) est un langage de modélisation orientée objet normalisé par l'OMG [1]. L'UML supporte un riche ensemble d'éléments de notation graphique et permet de modéliser une application selon une vision objet. Il décrit la notation pour les classes, les composants, les nœuds, les activités, le *workflow* [2], les cas d'utilisations, les objets, les états ainsi que la façon de modéliser les relations entre ces éléments. L'UML possède plusieurs facettes et permet de représenter un système selon différentes vues complémentaires : les diagrammes. Un diagramme UML est une représentation graphique qui s'intéresse à un aspect précis du modèle.

2.1 Le diagramme de packages

Les packages sont des éléments d'organisation des modèles. Ils regroupent des éléments de modélisation selon des critères logiques. Un diagramme de package est un diagramme de structure qui montre les packages et, éventuellement, les relations entre eux. Les relations les plus courantes entre différentes paquetages sont celles du type importation (import) et appel (call). Une importation de package est une relation entre un espace de nommage [3] et un package, qui signifie que l'espace de nommage ajoute tous les membres du package dans son propre espace de nommage. Une relation de dépendance d'utilisation du type appel (call) spécifie que la package invoque une opération du package cible.

1. L'OMG (Object Management Group) est une association américaine à but non-lucratif créée en 1989 dont l'objectif est de standardiser et promouvoir le modèle objet sous toutes ses formes. L'OMG est aussi à l'origine de la recommandation MDA (Model Driven Architecture, en anglais ou architecture dirigée par les modèles, en français).

2. Automatisation d'un processus (partiel ou complet), au cours duquel des documents, des informations et des tâches passent d'un participant à un autre, au sein d'un groupe de travail, en conformité avec un ensemble de règles prédéfinies. Un système de workflow définit, crée et gère l'exécution de tels processus.

3. Un espace de nommage (namespace en anglais) est un élément qui peut contenir en ensemble d'éléments nommés pouvant être identifiés par leurs noms. Un espace de nommage a la capacité d'importer des éléments individuels afin que ceux-ci puissent être référencés sans l'utilisation du nom qualifié à l'intérieur de l'espace de nommage importateur.

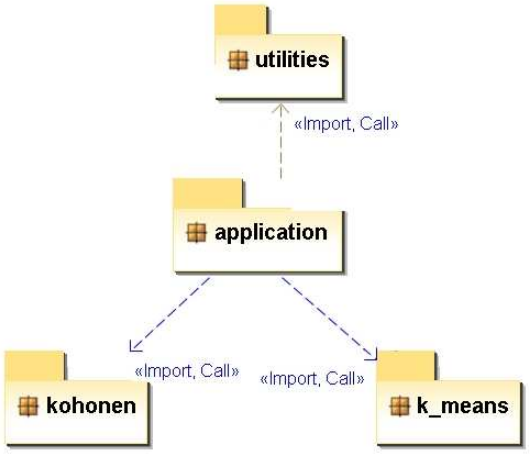

Figure 4.1 – Diagramme de packages de notre approche de classification de données évolutives.

Le diagramme de la figure 4.1 exhibe les packages de notre approche de classification de données évolutives. Ce diagramme présente un package principal appelé *application* où sont enregistres les classes de l'application principale (décrites dans la section 2.2 du présent chapitre). Les méthodes de classification appliquées étant indépendantes de notre approche, elles doivent être placées dans d'autres packages (dans notre exemple, les packages *k-means* ou *kohonen*) contenant des classes qui implémentent leurs fonctionnalités. Le package *application* peut ainsi faire appel à ces packages externes. De plus, les fonctionnalités additionnelles (par exemple, l'interaction avec le système d'exploitation, le formatage des données, etc.) utilisées doivent être implémentées en dehors du package *application*, dans un package réunissant les classes qui implémentent ces fonctionnalités (dans notre exemple, le package *utilities*).

2.2 Le diagramme de classes

Le diagramme de classes permet de fournir une représentation abstraite des objets du système. Chaque langage de Programmation Orienté Objets (POO) donne un moyen spécifique d'implémenter le paradigme objet (pointeurs, héritage multiple, etc.), mais le diagramme de classes permet de modéliser les classes du système et leurs relations indépendamment d'un langage de programmation particulier.

Le diagramme de la figure 4.2 exhibe les classes contenues dans le package *ap-*

plication. Sur cette figure, les relations de dépendance existantes sont du type :

- *Import* : Une relation de dépendance du type *import* indique que la classe source peut accéder aux éléments de la classe cible sans spécifier son nom qualificateur.
- *Call* : Une relation de dépendance d'utilisation du type *call* spécifie que la classe source invoque la classe cible ou une opération de la classe cible.
- *Instantiate* : Une relation de dépendance du type *instantiate* indique que la classe source crée des instances de la classe cible.

Le diagramme de la figure 4.2 contient 10 classes dont la principale est la classe centrale appelée *Application*. Dans cette classe sont implémentées, entre autres, des fonctionnalités pour la connexion avec la base de données, pour la lecture des données ainsi que des paramètres spécifiques de notre méthode, pour le découpage du tableau de données en fenêtres logiques ou temporelles , etc.

Dans la classe *NavigationStatistics* sont implémentées des fonctionnalités permettant la génération et le remplissage d'une table contenant des variables descriptives des navigations (cf. section 2.1 du chapitre 3).

La classe *PartitionComparaison* implémente des fonctionnalités liées au remplissage du tableau de contingence utilisé par les indices de comparaison de partition F-measure et indice corrigé de Rand.

La classe *WriteResult* a pour objectif d'implémenter des fonctions permettant l'enregistrement dans une base de données les résultats obtenus par notre approche pour chaque fenêtre analysée.

La classe *ArtificialData* a pour objectif l'implémentation des fonctions permettant la génération des données artificielles ainsi que la simulation de changements (cf. section 3 du chapitre 3).

La classe *Navigation* définit un nouveau type d'objet représentant une navigation d'internaute. Ainsi, une navigation enregistrée dans la base de données sur le disque sera chargée dans la mémoire centrale en tant qu'objet du type *Navigation*.

Dans la classe *SemanticChanges* sont implémentées des fonctionnalités liées à l'interprétation sémantique des changements repérés par notre approche. Cette classe implémente entre autres, l'algorithme d'interprétation sémantique des changements présenté dans la section 5.2 du chapitre 3.

La classe *Result* rassemble des informations liées au résultat de la méthode de classification, notamment les prototypes finaux, la valeur du critère optimisé, l'effectif des clusters et le label du cluster de chaque individu.

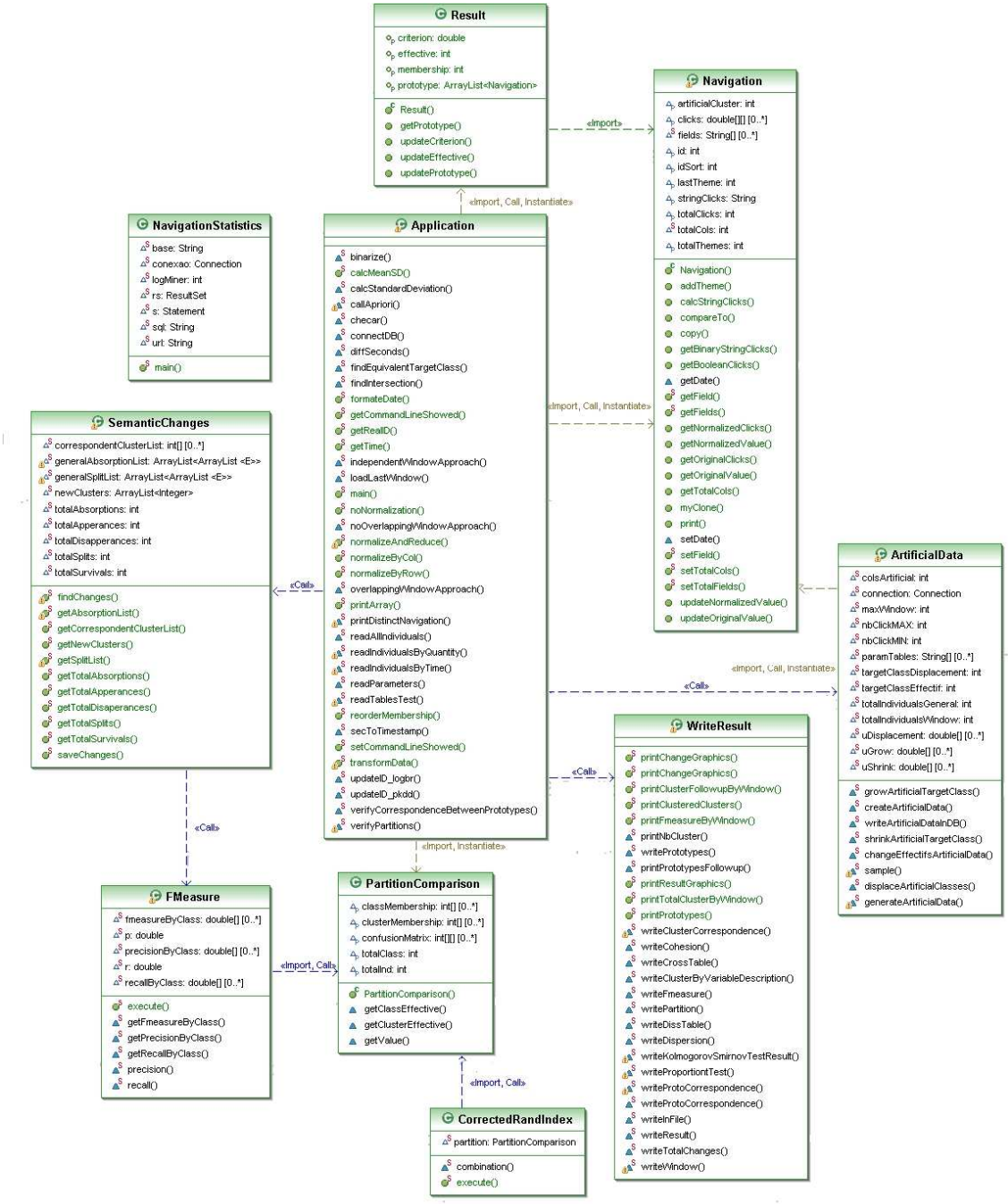

Figure 4.2 – Diagramme de classes contenues dans le package *application*.

3 Le modèle conceptuel des données

Le modèle conceptuel des données (MCD) a pour but de décrire de façon formelle les données qui seront utilisées par le système d'information. Il s'agit ainsi d'une représentation des données, facilement compréhensible, permettant de définir les dépendances ou relations entre ces différentes données. La figure 4.3 présente le schéma de l'entrepôt de données produit par notre approche de classification de données évolutives. Ce schéma continent un total de 12 tables, détaillées dans les paragraphes suivants.

La table *tb_ stat_ navigation* enregistre les variables descriptives des navigations (présentées dans la section 2.1 du chapitre 3). Le remplissage de cette table se fait de façon automatisée. Les requêtes SQL utilisées pour ce faire sont présentées dans la section 4 du présent chapitre.

La table *tb_ couting* enregistre les données du tableau de contingence analysé. Dans cette table, les champs du type c_i spécifient les thèmes de pages analysés. Dans cet exemple, les pages sont distribuées parmi 5 thèmes. Si nécessaire, plus de champs peuvent être rajoutés dans cette table afin de traiter les clics sur les nouveaux thèmes de pages.

Dans les deux tableaux cités ci-dessus, les individus sont enregistrés en ordre chronologique. Le champ *id_ navigation* assume des valeurs entières positives et représente l'ordre d'apparition des individus dans le tableau en question. Le champ $IDNavigation$ représente l'identifiant originel des navigations attribué pendant le processus de prétraitement. La valeur de ces deux champs peut être différente en fonction de l'application de filtres de sélection des navigations (par exemple, numéro minimum de clics, durée minimale, etc.).

La table *tb_ experiment* enregistre les paramètres à être utilisés lors des expérimentations. Les champs de cette table sont décrits dans le tableau 4.1. Il est important de remarquer qu'un même tableau de données (soit il décrit par des variables statistiques ou par le comptage des clics sur des thèmes de pages) peut faire l'objet de plusieurs expérimentations. De ce fait, il est possible d'appliquer notre approche de classification sur un même jeu de données tout en conservant le tableau de données cible (champ *tableau* de la table *tb_ experiment*) et faisant varier les paramètres de expérimentations (par exemple le type de fenêtrage, la taille de la fenêtre de données, etc.).

Une fois le tableau de données choisi et les paramètres de l'expérimentation

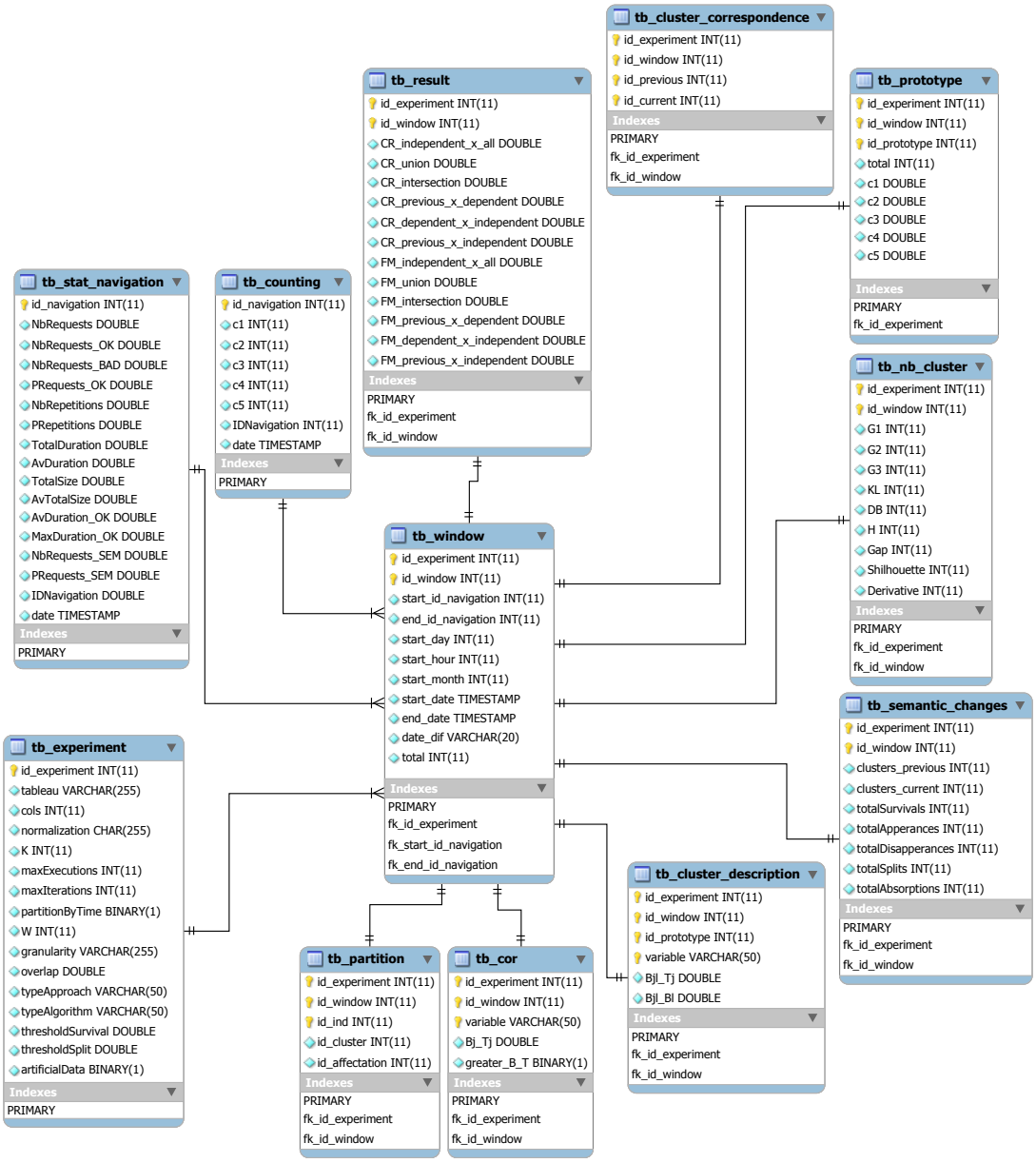

Figure 4.3 – Schéma de l'entrepôt de données produit par notre approche de classification de données évolutives.

définis, notre approche de classification peut être appliquée. Le tableau de données originel sera ainsi partitionné en fenêtres selon le type de fenêtrage choisi. La table *tb_window* enregistre des informations concernant les fenêtres engendrées par ce

Champ	Description
id_experiment	identifiant de l'expérimentation
tableau	nom du tableau contenant les données à analyser
cols	dimension de l'espace de données
normalization	type de normalisation à être appliquée
K	nombre de classes demandées
maxExecutions	nombre maximum de répétitions de la méthode de classification
maxIterations	nombre maximum d'itérations de la méthode de classification
partitionByTime	spécifie le type de fenêtrage (0 : logique, 1 : temporelle)
W	taille de la fenêtre (soit en nombre d'effectifs, soit en durée de temps)
granularity	si le fenêtrage temporal est appliqué, ce champ spécifie la granularité (heure, jour, semaine, mois, etc.) de la durée de temps spécifiée dans le champ W
overlap	si une zone de chevauchement entre les fenêtres voisines est demandée, ce champ indique le pourcentage de recouvrement souhaité
typeApproach	spécifie la stratégie de classification
typeAlgorithm	spécifie la méthode de classification
thresholdSurvival	seuil indiquant le pourcentage d'individus à être conservés par un cluster survivant
thresholdSplit	seuil indiquant le pourcentage d'individus à être conservés par un cluster issu d'une scission
artificialData	spécifie si les données traitées sont réelles ou artificielles (0 : réelles, 1 : artificielles)

Table 4.1 – Description des champs de la table *tb_experiment*.

partitionnement. Les champs de cette table sont décrits dans le tableau 4.2.

Chaque fenêtre est associée à un tableau de données ainsi qu'à une expérimentation. L'identifiant unique de chaque fenêtre est une combinaison de deux champs : *(i)* l'identifiant de l'expérimentation à laquelle est associée la fenêtre et *(ii)* un entier positif identifiant uniquement chaque fenêtre dans l'expérimentation concernée.

Dans la table *tb_window*, les champs *start_id_navigation* et *end_id_navigation* identifient le sous-ensemble du tableau de données originel traité par la fenêtre. Ces deux champs représentent respectivement le premier et le dernier indice des navigations enveloppées dans chaque fenêtre.

Après l'application de la méthode de classification sur les individus d'une fenêtre, plusieurs informations en sont engendrées et doivent être enregistrées dans la base de données. Ces informations concernent une seule fenêtre de données à la fois et sont stockées dans les autres tables de l'entrepôt de données. Ces tables héritent l'identifiant unique de la fenêtre (*id_experiment* + *id_window*) et ajoutent, selon besoin, d'autres identifiants propres.

La table *tb_result* enregistre les valeurs obtenues par les indices de comparaison de partitions (F-measure et indice corrige de Rand) pour chaque fenêtre analysée. Dans cette table sont discriminées les valeurs des indices obtenues lors de la comparaison des partitionnes obtenues par les différentes stratégies de classification.

La table *tb_nb_cluster* enregistre le nombre de clusters suggéré pour chaque indice de détermination du nombre de clusters (cf. section 4 du chapitre 2) pour

Champ	Description
id_experiment	identifiant de l'expérimentation à laquelle la fenêtre est attachée
id_window	identifiant de la fenêtre
start_id_navigation	identifiant du premier individu dans la fenêtre
end_id_navigation	identifiant du dernier individu dans la fenêtre
start_day	jour du premier clic dans la fenêtre
start_hour	heure du premier clic dans la fenêtre
start_month	mois du premier clic dans la fenêtre
start_date	date de début de la fenêtre
end_date	date de fin de la fenêtre
date_dif	différence entre la date de début et de fin de la fenêtre
total	total d'individus contenus dans la fenêtre

Table 4.2 – Description des champs de la table *tb_window*.

Champ	Description
id_experiment	identifiant de l'expérimentation à laquelle la fenêtre est attachée
id_window	identifiant de la fenêtre
clusters_previous	nombre de clusters dans la fenêtre immédiatement précédente
clusters_current	nombre de clusters dans la fenêtre courante
totalSurvivals	nombre de clusters survivants
totalApperances	nombre de clusters apparus
totalDisapperances	nombre de clusters disparus
totalSplits	nombre de clusters scissionnés
totalAbsorptions	nombre de clusters fusionnés ou absorbés

Table 4.3 – Description des champs de la table *tb_semantic_changes*.

chaque fenêtre analysée.

La table *tb_partition* concerne les partitions obtenues dans chaque fenêtre. Dans cette table sont enregistrés, pour chaque individu (champ *id_ind*), sa classe d'affectation (champ *id_affectation*) dans la partition P_1^t ainsi que la classe de la partition P_2^t à laquelle il appartient après l'application de la méthode de classification sur les données de la fenêtre courante (champ *id_cluster*).

La table *tb_prototype* enregistre les prototypes fournis par la méthode de classification.

La table *tb_semantic_changes* enregistre l'interprétation sémantique des changements détectés par notre approche. Les champs de cette table sont décrits dans te tableau 4.3.

La table *tb_cluster_correspondance* enregistre, pour chaque cluster dans la partition P_1^t (champ *id_previous*) le cluster correspondant dans la partition P_2^t (champ *id_current*).

Les deux dernières tables enregistrent des informations liées aux critères d'interprétation des clusters par leurs variables les plus discriminantes (cf. section 6 du chapitre 2) pour chaque fenêtre analysée.

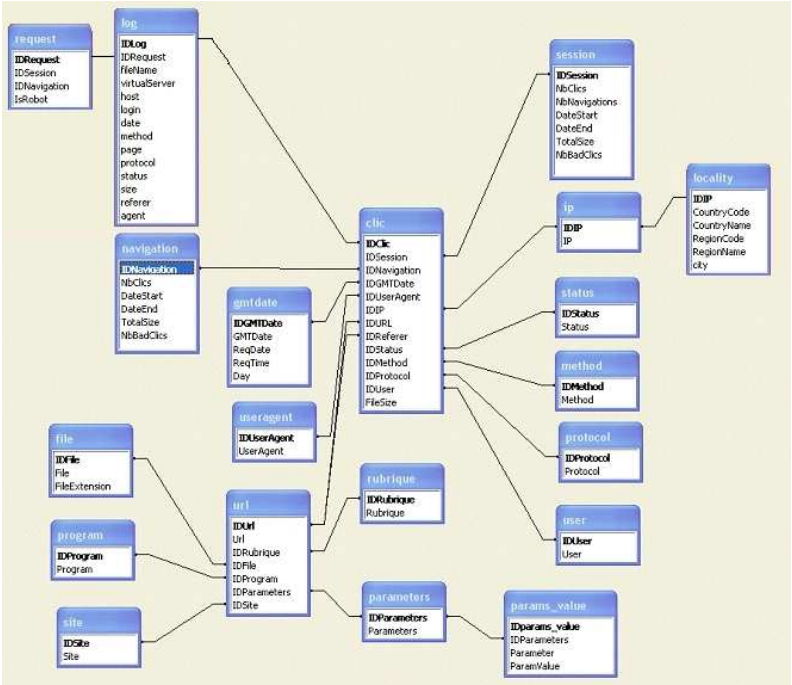

Figure 4.4 – Schéma en étoile de l'entrepôt de données produit par l'outil de prétraitement de [8].

4 Extraction des variables descriptives des navigations

Dans cette section, nous allons détailler les requêtes SQL utilisées pour l'extraction des variables statistiques descriptives des navigations. Après l'application de l'outil de prétraitement proposé par [8], un entrepôt de données en schéma en étoile est généré (cf. figure 4.4, extraite de [8]). Sur cette base de données relationnelle, nous appliquons une série de requêtes SQL afin de peupler la table *tb_stat_navigation* du schéma de la figure 4.3.

Le script suivant génère la table *tb_stat_navigation* contenant toutes les variables en question. Les navigations sont ordonnées selon la date de leur premier clic. Les navigations placées dans cette table doivent respecter certaines contraintes, et à cause de cela, le champ *id* du type auto-incrément est utilisé comme identifiant unique de chaque navigation. L'identifiant original des navigations est enregistré sur le champ *IDNavigation*.

```
1 CREATE TABLE tb_stat_navigation (
```

```
2    id DOUBLE AUTO_INCREMENT,
3    NbRequests DOUBLE NOT NULL,
4    NbRequests_OK DOUBLE NOT NULL,
5    NbRequests_BAD DOUBLE NOT NULL,
6    PRequests_OK DOUBLE NOT NULL,
7    NbRepetitions DOUBLE NOT NULL,
8    PRepetitions DOUBLE NOT NULL,
9    TotalDuration DOUBLE NOT NULL,
10   AvDuration DOUBLE NOT NULL,
11   TotalSize DOUBLE NOT NULL,
12   AvTotalSize DOUBLE NOT NULL,
13   AvDuration_OK DOUBLE NOT NULL,
14   MaxDuration_OK DOUBLE NOT NULL,
15   IDNavigation DOUBLE NOT NULL,
16   date TIMESTAMP NOT NULL,
17 PRIMARY KEY ( id, IDNavigation), INDEX (IDNavigation)
18 );
```

En exécutant le script suivant, nous peuplons la table *tb_stat_navigation* avec des navigations respectant certaines contraintes, par exemple, nombre minimum de 10 clics, durée minimale de 1 minute, vitesse de navigation maximale égale à 15 clics par minute.

```
1 INSERT INTO tb_stat_navigation (IDNavigation, NbRequests, TotalDuration,
2 AvDuration, TotalSize, AvTotalSize)
3 SELECT n.IDNavigation, n.NbClics, (n.DateEnd - n.DateStart)/1000,
4 ((n.DateEnd - n.DateStart)/1000*n.NbClics), n.TotalSize,
5 (n.TotalSize/n.NbClics)
6 FROM navigation n
7 WHERE n.NbClics >= 10 AND ((n.DateEnd - n.DateStart)/1000) >= 60
8 AND ((n.DateEnd - n.DateStart)/1000*n.NbClics) >= 4
9 ORDER BY n.datestart;
```

Dans le schéma de la figure 4.4, la date est enregistrée sous le format universel, c'est-à-dire, le nombre de millisecondes écoulées depuis minuit du 01/01/1970. Pour obtenir la date dans un format 'AAAA-MM-JJ HH :MM :SS', il est nécessaire appliquer la fonction FROM_UNIXTIME de SQL. Le script suivant met à jour le champ *date* de la table *tb_stat_navigation*.

updates date

```
1 UPDATE tb_stat_navigation rn, navigation n
2 SET rn.date = from_unixtime(n.datestart/1000)
3 WHERE rn.IDNavigation = n.IDNavigation;
```

Le script suivant rempli le champ *NbRepetitions* de la table *tb_stat_navigation*. Pour ce faire, une table auxiliaire nommée *temp* est créé. Cette table a pour but de compter le nombre de pages ayant un seul clic dans chaque navigation. Pour chaque navigation, le nombre de pages répétées sera égal à la différence entre le nombre

total de clics et le nombre de pages ayant un seul clic.

```
1 CREATE TABLE temp
2 SELECT n.IDNavigation, COUNT( DISTINCT (u.URL) ) AS total_distinct_pages
3 FROM clic l, tb_stat_navigation n, url u
4 WHERE l.IDNavigation = n.IDNavigation AND l.IDURL = u.IDURL
5 GROUP BY n.IDNavigation;
6 ALTER TABLE temp ADD PRIMARY KEY (IDNavigation);
7 UPDATE tb_stat_navigation, temp
8 SET tb_stat_navigation.NbRepetitions =
9 (tb_stat_navigation.NbRequests - temp.total_distinct_pages)
10 WHERE tb_stat_navigation.IDNavigation = temp.IDNavigation;
11 DROP TABLE temp;
```

Une fois le champ *NbRepetitions* rempli, le pourcentage de clics répétés dans une navigation est facile à calculer, comme le montre le script suivant.

```
1 UPDATE tb_stat_navigation
2 SET PRepetitions = (NbRepetitions/NbRequests);
```

Le script suivant rempli le champ *NbRequests_OK*. Une requête réussie étant identifiée par le code statut égal à 200, la table auxiliaire *temp* est utilisée afin de compter le nombre de clics réussis dans chaque navigation.

```
1 CREATE TABLE temp
2 SELECT n.IDNavigation, COUNT(l.IDClic) as c
3 FROM tb_stat_navigation n, clic l, status s
4 WHERE n.IDNavigation = l.IDNavigation AND l.IDStatus = s.IDStatus AND
5 s.status = 200
6 GROUP BY n.IDNavigation;
7 ALTER TABLE temp ADD PRIMARY KEY (IDNavigation);
8 UPDATE tb_stat_navigation, temp
9 SET NbRequests_OK = temp.c
10 WHERE tb_stat_navigation.IDNavigation = temp.IDNavigation;
11 DROP TABLE temp;
```

Le script suivant rempli le champ *NbRequests_BAD*. Une requête échouée étant identifiée par le code statut autre que 200, la table auxiliaire *temp* est utilisée afin de compter le nombre de clics échoués dans chaque navigation.

```
1 CREATE TABLE temp
2 SELECT n.IDNavigation, COUNT(l.IDClic) as c FROM tb_stat_navigation n,
3 clic l, status s
4 WHERE n.IDNavigation = l.IDNavigation AND l.IDStatus = s.IDStatus
5 AND s.status <> 200
6 GROUP BY n.IDNavigation;
7 ALTER TABLE temp ADD PRIMARY KEY (IDNavigation);
8 UPDATE tb_stat_navigation, temp
9 SET NbRequests_bad = temp.c
10 WHERE tb_stat_navigation.IDNavigation = temp.IDNavigation;
11 DROP TABLE temp;
```

Une fois le champ *NbRequests_ OK* rempli, le pourcentage de clics réussis dans une navigation est facile à calculer, comme le montre le script suivant.

```
1 UPDATE tb_stat_navigation
2 SET PRequests_OK = (NbRequests_OK/NbRequests);
```

Les deux scripts suivants ont pour but de remplir les champs *MaxDuration_ OK* et *AvDuration_ OK*. Pour ce faire, nous utilisons deux tables auxiliaires temp1 et temp2. Dans temp1, nous allons ordonner chronologiquement les clics dans chaque navigation. Dans temp2, nous allons enregistrer la durée de chaque clic, calculée en fonction du nombre de secondes passées entre un clic et le clic immédiatement suivant. Le champ *MaxDuration_ OK* de chaque navigation sera rempli avec la durée du clic réussi le plus long.

```
1 CREATE TABLE temp1
2 SELECT l.IDNavigation, l.IDClic , d.IDGMTDate, d.GMTDate as d,
3 FROM_UNIXTIME(d.GMTDate/1000) as date
4 FROM clic l, gmtdate d
5 WHERE l.IDGMTDate = d.IDGMTDate
6 GROUP BY l.IDnavigation, l.IDClic , d.IDGMTDate
7 ORDER BY l.IDnavigation, l.IDClic , d.IDGMTDate;
8 ALTER TABLE temp1 ADD INDEX (IDnavigation, IDClic, IDGMTDate);
9 CREATE TABLE temp2
10 SELECT a.IDnavigation, a.IDClic,
11   (SELECT MIN(b.d)
12   FROM temp1 b
13   WHERE b.d > a.d AND a.IDnavigation = b.IDnavigation) - a.d AS Duration
14 FROM temp1 a
15 ORDER BY a.IDnavigation, a.IDClic;
16 ALTER TABLE temp2 ADD PRIMARY KEY (IDNavigation,IDClic);
17 UPDATE tb_stat_navigation rn,
18   (SELECT t2.IDNavigation, MAX(t2.Duration) AS m
19   FROM temp2 t2, status s, clic l
20   WHERE l.IDStatus = s.IDStatus AND status = 200 AND t2.IDNavigation = l.
       IDNavigation
21   GROUP BY t2.IDNavigation) t1
22 SET MaxDuration_OK = t1.m/1000 WHERE rn.IDNavigation = t1.IDNavigation;
```

Ensuite, le champ *AvDuration_ OK* sera rempli avec la durée moyenne des clics réussis.

```
1 UPDATE tb_stat_navigation rn,
2   (SELECT t2.IDNavigation, AVG(t2.Duration/1000) as m
3   FROM temp2 t2, status s, clic l
4   WHERE l.IDStatus = s.IDStatus AND status = 200 AND t2.IDNavigation = l.
       IDNavigation
5   GROUP BY t2.IDNavigation) t1
6 SET AvDuration_OK = t1.m
7 WHERE rn.IDNavigation = t1.IDNavigation;
8 DROP TABLE temp2, temp1;
```

En ce qui concerne les sites Web ayant une structure sémantique de pages, les champs *NbRequests_SEM* et *PRequests_SEM* sont prévus pour enregistrer le nombre et le pourcentage de clics liés aux pages de cette structure. Le script suivant intègre ces deux champs à la fin de la table *tb_stat_navigation*.

```
1 ALTER TABLE tb_stat_navigation
2 ADD NbRequests_SEM DOUBLE NOT NULL AFTER MaxDuration_OK,
3 ADD PRequests_SEM DOUBLE NOT NULL AFTER NbRequests_SEM;
```

L'un des sites analysés pendant nos expérimentations utilise des servlets Java pour la génération automatique des pages liées à sa structure sémantique. Les URL's concernant les clics sur les pages de cette structure sont du type : `http://notitia.cin.ufpe.br/servlets/newstorm.notitia.apresentacao.ServletDeSecao?codigoDaSecao=26&dataDoJornal=atual`

Chaque page de la structure sémantique est identifiée par un code servlet unique. Ce code se trouve toujours entre la chaine de caractères *'codigoDaSecao='* et le caractère &. Dans l'exemple, le code de la page en question est 26. Ceci étant vérifié, le script suivant crée une table nommée *tb_code_servlet* qui enregistre, pour chaque clic lié à la structure sémantique, le code de la page concernée.

```
1 CREATE TABLE tb_code_servlet
2 SELECT IDURL, url,
3 substring_index( substring_index( url, 'codigoDaSecao=', -1 ) , '&', 1 ) AS
      code
4 FROM url WHERE url LIKE '\%codigoDaSecao=\%';
5 ALTER TABLE tb_code_servlet ADD PRIMARY KEY (IDURL) ;
6 DELETE FROM tb_code_servlet WHERE CONVERT(code, SIGNED INTEGER) = 0;
```

Le script suivant met à jour le champ *NbRequests_SEM*. Pour ce faire, la table temporaire *temp* est utilisée pour compter le nombre total de clics liés aux pages présentes dans la structure sémantique du site en question.

```
1 UPDATE tb_stat_navigation rn,
2   (SELECT l.IDNavigation, COUNT(*) AS total
3    FROM clic l, tb_code_servlet c
4    WHERE l.IDURL = c.IDURL
5    GROUP BY l.IDNavigation) temp
6 SET NbRequests_SEM = temp.total
7 WHERE rn.IDNavigation = temp.IDNavigation;
```

Une fois le champ *NbRequests_SEM* rempli, déterminer le pourcentage de clics liés aux pages présentes dans la structure sémantique devient facile, comme montre le script suivant.

```
1 UPDATE tb_stat_navigation
2 SET PRequests_SEM = (NbRequests_SEM/NbRequests);
```

5 Synthèse

Dans ce chapitre, nous avons présenté la modélisation de notre application à l'aide du formalisme UML. Ce langage fournit des outils permettant la modélisation des parties intégrantes d'un système, et ce indépendamment d'un langage de programmation particulier.

Pour l'extraction des variables statistiques descriptives des navigations, nous avons eu recours au formalisme du langage de manipulation des données SQL, très répandu et de compréhension facile. Les requêtes SQL utilisées ont été délibérément détaillées afin de faciliter leur reproduction.

Chapitre 5

Expérimentations et analyse des résultats

1 Introduction

Le présent chapitre a pour but de décrire les expérimentations réalisées sur des données artificielles (détaillées dans la section 2) produites par notre méthodologie de génération de données (présentée dans la section 3 du chapitre 3) ainsi que les expérimentations réalisées sur des données réelles issues de différents domaines d'application (détaillées dans la section 3). Pour chacune de ces expérimentations, nous présentons le jeu de données utilisé et nous analysons les résultats obtenus après l'application de notre approche de classification pour la détection et le suivi des changements sur des données évolutives. Finalement, dans la section 4, nous décrivons les expérimentations réalisées sur les différents indices de détermination du nombre de clusters sur des jeux de données réelles et artificielles.

2 Expérimentations sur des données artificielles

Afin de vérifier l'efficacité de notre approche de classification pour la détection et le suivi des changements sur des données évolutives, nous avons réalisé trois études de cas sur des jeux de données simulées ayant différents dégrés de complexité.

La première étude de cas traite le changement d'effectifs sur un jeu de données contenant des classes bien séparées, la seconde étude concerne le changement d'effectifs sur un jeu de données contenant des classes recouvrantes. Finalement, la

troisième étude aborde l'effet de rapprochement entre les classes bien séparées. Notons que lors du déplacement des classes dans la troisième étude de cas, les effectifs des classes restent constants.

Pour ces trois études, les expérimentations ont été réalisées sur un jeu de données artificiellement généré contenant cinq classes. Au départ, les effectifs des classes sont équiprobables et égales à 20% de la taille de la population, ce qui donne un total de 2 000 individus par classe pour une fenêtre contenant un total de 10 000 individus. Le tableau 5.1 présente les valeurs des paramètres utilisés pour la génération des données artificielles à partir de l'application de la méthodologie présentée dans la section 3 du chapitre 3. Dans ce tableau, p correspond au nombre de variables (et peut représenter, par exemple, le nombre de thèmes de pages d'un site Web fictif), $nbxWindow$ représente le nombre de fenêtres à générer, $totalIndividuals$ représente le nombre d'individus par fenêtre, $c^{'}$ représente la classe cible des changements et les variables du type β_i représentent les facteurs de changements (définis dans la section 3 du chapitre 3).

Paramètre	Valeurs utilisées lors des expérimentations
p	10
nbWindow	2
nbClicMIN	10
nbClicMAX	50
totalIndividuals	10 000
$c^{'}$	1 (changement d'effectifs) et 2 (déplacement)
β_1 (facteur de rétrécissement)	$\{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$
β_2 (facteur de grossissement)	$\{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$
β_3 (facteur de déplacement)	$\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$

Table 5.1 – Valeurs des paramètres utilisées lors des expérimentations sur des données artificielles.

Nous avons adopté un découpage en fenêtres logiques de taille égale à 10 000 (cf. tableau 5.1) et comme méthode de classification de données, nous avons utilisé la MND [40] [42] initialisée avec des prototypes aléatoires et appliquée avec la distance Euclidienne sur les données de chaque fenêtre. Le nombre K de classes étant connu à l'avance et égal à 5.

Les tableaux 5.2 et 5.3 contiennent respectivement les prototypes initiaux (noyaux) utilisés pour la génération des classes bien séparées et recouvrantes. Une projection des classes sur le premier plan factoriel d'une ACP est présentée sur la figure 5.1. Sur cette figure, les classes sont numérotées par un label positionné sur leur centre

Table 5.2 – Noyaux des classes bien séparées.

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
c_1	0.30	0.00	0.25	0.00	0.00	0.10	0.00	0.25	0.00	0.10
c_2	0.00	0.35	0.00	0.00	0.25	0.00	0.00	0.00	0.40	0.00
c_3	0.10	0.00	0.00	0.20	0.00	0.30	0.00	0.00	0.00	0.40
c_4	0.00	0.00	0.18	0.00	0.36	0.00	0.20	0.00	0.26	0.00
c_5	0.00	0.22	0.00	0.32	0.00	0.16	0.00	0.25	0.00	0.05

de gravité, les points sont liés aux centres de leurs classes par un trait et les cercles sont reliés aux indices de leurs classes.

Table 5.3 – Noyaux des classes recouvrantes.

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
c_1	0.00	0.20	0.05	0.09	0.12	0.20	0.00	0.00	0.18	0.16
c_2	0.10	0.10	0.10	0.00	0.10	0.00	0.18	0.12	0.12	0.18
c_3	0.15	0.20	0.16	0.20	0.00	0.22	0.00	0.00	0.00	0.07
c_4	0.20	0.10	0.00	0.10	0.20	0.12	0.10	0.05	0.13	0.00
c_5	0.00	0.22	0.00	0.16	0.16	0.16	0.05	0.20	0.00	0.05

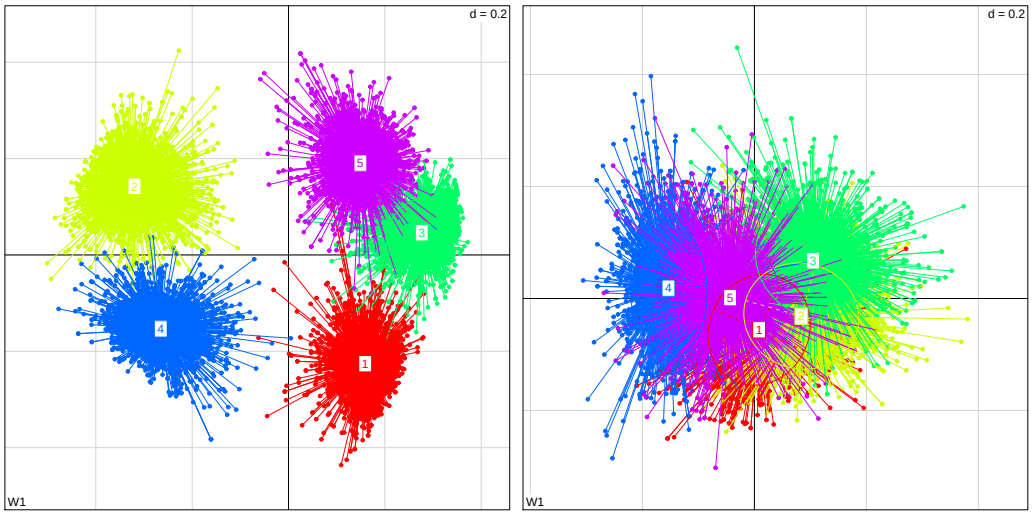

Figure 5.1 – Projection de cinq classes artificielles bien séparées (à gauche) et de cinq classes artificielles recouvrantes (à droite).

Après la génération des individus dans la première fenêtre, nous avons simulé les effets de grossissement et de rétrécissement des classes (cf. section 3.1 du chapitre 3) sur une deuxième fenêtre de données. Pour ce faire nous avons adopté la classe

1 en tant que classe cible des changements. Nous avons également simulé, dans la deuxième fenêtre, l'effet de déplacement des classes vers une classe cible (cf. section 3.2 du chapitre 3), dans notre exemple la classe 2. Les valeurs assumées par ces facteurs de changements sont précisées dans le tableau 5.1. Pour le cas des changements liés à l'effectif des classes, plus petite la valeur des facteurs β_1 et β_2, plus intense est le changement simulé (rétrécissement et grossissement de la classe cible). Pour le cas de déplacement de classes, plus grand la valeur du facteur β_3, plus proches de la classe cible sont les classes déplacées.

Par la suite, nous avons appliqué notre approche de détection et suivi des changements sur ces deux fenêtres de données afin de vérifier sa capacité de repérer les changements préétablis. A l'aide des critères de validation externe F-measure et indice corrigé de Rand (présentés dans la section 5 du chapitre 2), nous avons comparé les deux partitions P_1^t et P_2^t obtenues sur les individus de la deuxième fenêtre (comme expliqué dans la section 5 du chapitre 3), c'est-à-dire, la partition P_1^t est obtenue par l'affectation des individus de la deuxième fenêtre aux prototypes finaux issus de la classification effectuée sur la première fenêtre. La partition P_2^t est celle obtenue après la convergence de la méthode de classification initialisée avec des prototypes aléatoires sur les données de la deuxième fenêtre.

Les sections suivantes présentent les résultats obtenus pour chaque étude de cas.

2.1 Étude de cas I : les classes sont bien séparées

La courbe des valeurs obtenues par la F-measure et par l'indice corrigé de Rand (CR) est présentée sur le graphique à gauche de la figure 5.2. Sur ce graphique, l'axe des ordonnées présente les valeurs des deux indices de validation et l'axe des abscisses présente les valeurs des facteurs de rétrécissement β_1 (représenté par r) et de grossissement β_2 (représenté par g).

Notons que les valeurs des indices de comparaison de partitions varient selon la valeur des facteurs de rétrécissement et grossissement. Plus petite est la valeur des facteurs de changement, plus petites sont les valeurs obtenues par les indices de comparaison de partitions.

Comme les changements les plus importants sont provoqués par les valeurs les plus petites des facteurs de changement, et comme les valeurs les plus petites des indices de comparaison indiquent un niveau plus accentué de désaccord entre les deux partitions comparées, nous pouvons conclure que l'approche adoptée est capable de

repérer les changements préétablis insérés sur les données artificielles.

Notons également que l'indice corrigé de Rand est plus sensible aux changements et présente toujours des valeurs inférieures à celles obtenues par la F-measure. Ceci est dû au fait que l'indice corrigé de Rand utilise une analyse globale des deux partitions comparées, alors que la F-measure réalise une analyse plus fine cluster par cluster.

Les changements deviennent plus importants quand les facteurs de changement assument de valeurs inférieures à 0.8 aussi bien lors du grossissement que du rétrécissement de la classe cible. Un autre fait important à remarquer est que le grossissement de la classe cible implique plus de changement au niveau de la partition que le rétrécissement de la même classe.

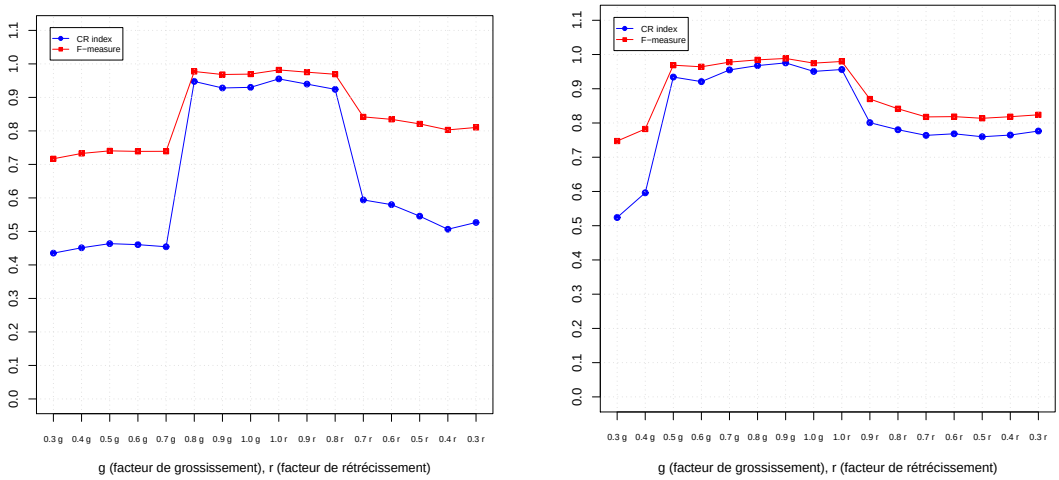

Figure 5.2 – Valeurs de la F-measure et de l'indice corrigé de Rand (CR) pour les classes artificielles bien separées (à gauche) et pour les classes artificielles recouvrantes (à droite).

2.2 Étude de cas II : les classes sont recouvrantes

Pour le cas des classes recouvrantes, les valeurs obtenues par les mêmes indices de comparaison de partition son présentées sur le graphique à droite de la figure 5.2. Comme nous pouvons remarquer, le niveau de changements obtenu par les deux indices de comparaison est moins important que celui issu de l'analyse sur

les classes bien séparées. Cela est dû au fait que les classes recouvrantes utilisées dans cet exemple ont beaucoup de zones de chevauchement et représentent ainsi une difficulté de classification plus importante.

Malgré le niveau de difficulté de classification intrinsèque à ce jeu de données, les indices de comparaison ont tout de même été capables de repérer les changements préétablis.

Comme dans le cas des classes bien séparées, les changements les plus importants sont provoqués par les valeurs les plus petites des facteurs de changement. Pour le grossissement de la classe cible, les changements deviennent plus importants pour les valeurs du facteur de grossissement inferieures à 0.5, alors que les changements au niveau du rétrécissement de la classe cible sont plus importants pour des valeurs inferieures à 1.0.

2.3 Étude de cas III : les classes se déplacent

Pour la simulation de l'effet de déplacement des classes dans l'espace de données, nous avons utilisé le jeu de données contenant les cinq classes bien séparées présentées à gauche de la figure 5.1. La classe cible adoptée est la classe numéro 2. En variant les valeurs du facteur de déplacement β_3 (cf. tableau 5.1), l'effet obtenu lors du déplacement des classes est illustré sur la figure 5.3. Plus élevées sont ces valeurs, plus proches de la classe cible sont les classes déplacées.

Les valeurs obtenues par les indices de comparaison de partitions sont présentées sur le graphique de la figure 5.4. Sur ce graphique, l'axe des ordonnées représente les valeurs des indices de comparaison et l'axe des abscisses représente les valeurs du facteur de déplacement β_3 (représenté par d). Comme nous pouvons remarquer, les valeurs obtenues par les deux indices décroissent au fur et à mesure que les valeurs du facteur de déplacement augmentent. Ceci est dû au fait que les valeurs plus élevées du facteur de déplacement provoquent un plus grand niveau de rapprochement des classes (et donc de changements) au regard de la position initiale des classes dans la première fenêtre.

Ce graphique démontre clairement le potentiel que les deux indices adoptés ont de repérer le niveau de changements présents entre les deux partitions comparées. Encore une fois, l'indice corrigé de Rand se montre plus sensible aux changements que la F-measure.

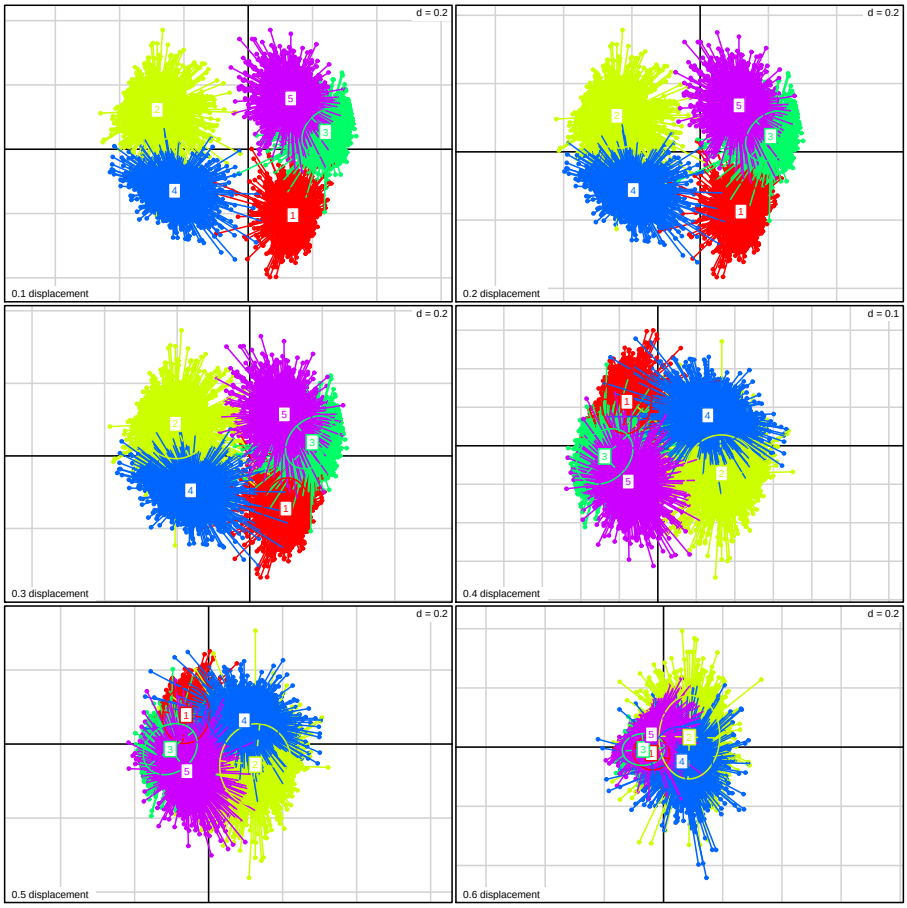

Figure 5.3 – Cinq classes artificielles bien séparées dont quatre se déplacent vers la classe cible (labelisée 2). La valeur au bas à gauche de chaque graphique représente la valeur utilisée pour le facteur de déplacement.

3 Expérimentations sur des données réelles

En ce qui concerne la distance de dissimilarité et la méthode de classification adoptées pour les expérimentations décrites dans cette section, nous appliquons la distance euclidienne combinée soit (i) avec la MND [40] [42] initialisée avec des prototypes aléatoires et le nombre de clusters fixé *a priori*, soit (ii) avec la méthode des cartes auto-organisatrices de kohonen (SOM) [57]. Dans ce dernier cas, la carte initiale contient approximativement 200 neurones distribués sur une topologie rectangulaire et initialisés à partir d'une analyse en composantes principales (comme expliqué dans la section 3.2 du chapitre 2) appliquée sur les données contenues à l'intérieur de chaque fenêtre analysée. Par la suite, une classification ascendante hié-

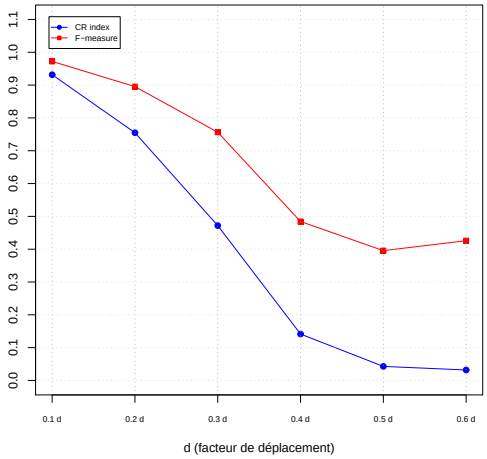

Figure 5.4 – Valeurs de la F-measure et de l'indice corrigé de Rand pour les classes artificielles déplaçantes.

rarchique utilisant le critère de Ward et la distance Euclidienne (cf. section 2.1 du chapitre 2) est appliquée sur les prototypes finaux fournis par la SOM. Le dendrogramme obtenu est coupé selon le critère basé sur les dérives premières et secondes (décrit dans la section 4.2 du chapitre 2). Ce processus fournit un nombre de clusters de façon automatisée.

Tous les jeux de données traitées dans le cadre de nos analyses ont été centrées et réduites avant l'application de la méthode de classification.

3.1 Étude de cas IV : analyse d'un site Web académique

Comme première étude de cas sur des données réelles, nous analysons les fichiers log du site Web du Centre d'Informatique (CIn) de l'Université Fédérale de Pernambuco (UFPE) à Recife, Brésil [1]. La figure 5.5 exhibe un aperçu du site en question. Ce site est constitué d'un ensemble de pages statiques (pages personnelles, pages de support de cours, etc.) et dynamiques, ces dernières sont gérées par des *servlets* programmées en Java. La partie dynamique du site consiste en 90 pages bien organisées sous la forme d'un arbre sémantiquement structuré, présenté sur la figure 5.6. Les pages de la structure sémantique sont présentées à gauche du site en guise de menu expansible.

1. Le site analysé est accessible à l'adresse suivante : `http://www.cin.ufpe.br/`

Figure 5.5 – Aperçu de la page de garde du site Web du centre d'informatique de l'UFPE-Brésil.

Pendant nos expérimentations, nous avons étudié les accès au site du 01 juillet 2002 à 00 :10 :25 jusqu'au 15 mai 2003 à 13 :22 :27. L'ensemble de fichiers log Web concernés ayant une taille d'environ 2 gigaoctets. Afin d'analyser les traces d'usage les plus représentatives, nous avons sélectionné les navigations longues (contenant au moins 10 requêtes et avec une durée totale d'au moins 1 minute) et supposées humaines (le ratio de la durée sur le nombre de requêtes doit être supérieur à 4, c'est-à-dire, correspondre à moins de 15 requêtes par minute). Ces efforts ont pour but d'extraire les navigations originées par des êtres humains et d'exclure celles provenant des robots. L'élimination des navigations courtes est motivée par la recherche de patterns d'utilisation du site, à l'exclusion des accès simples (par l'intermédiaire d'un moteur de recherche, par exemple) qui n'engendre pas de parcours sur le site. Après l'application de ces contraintes, l'ensemble de données à analyser continent un total de 138 536 navigations. Le tableau de données d'entrée utilisé est défini par des variables statistiques sur les navigations (décrites dans le tableau 3.1 du chapitre 3).

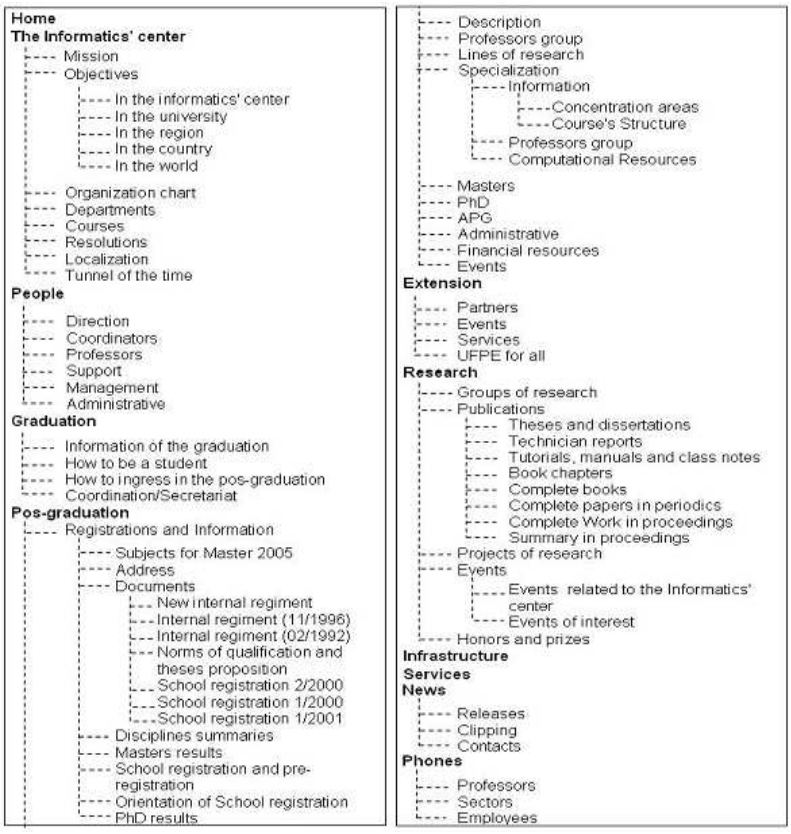

Figure 5.6 – Structure sémantique du site Web du centre d'informatique de l'UFPE-Brésil.

Analyse des résultats obtenus par les différentes stratégies de classification appliquées sur le tableau de variables statistiques descriptives des navigations

Dans cette section, nous allons analyser les résultats obtenus par les différentes stratégies de classification non supervisée (proposées dans la section 4 du chapitre 3) appliquées avec la méthode des Nuées Dynamiques et la distance Euclidienne (cf. section 4.3 du chapitre 3).

Pour les expérimentations avec ces stratégies de classification, nous avons fixé le nombre de clusters K égal à 10 et le nombre d'initialisations aléatoires de la méthode de classification égal à 50, hormis le cas où l'algorithme est initialisé à partir des prototypes issus de la classification sur les données de la fenêtre immédiatement précédente (classifications locales *précédente* (CL_2) et *dépendante* (CL_3)).

Pour le partitionnement de l'ensemble de données d'usage en question, nous

avons adopté des fenêtre temporelle de taille égale à 1 mois.

Pour mieux comprendre l'évolution des clusters sur les fenêtres de temps analysées, nous avons réalisé un suivi des prototypes des clusters mois par mois pour la classification locale *indépendante* (CL_1) et la classification locale *dépendante* (CL_3). Nous avons projeté ces prototypes sur le premier plan factoriel d'une ACP (voir figure 5.7). Sur ce graphique, chaque point noir représente un individu (navigation), chaque cercle coloré représente un prototype.

Pour le cas de la classification locale *dépendante* (à droite de la figure), les prototypes des dix clusters sont représentés par des couleurs différentes. Pour cette stratégie de classification, on note une certaine stabilité malgré la diversité des mois analysés.

Pour le cas de la classification locale *indépendante* (à gauche de la figure), la trajectoire temporelle est simplement matérialisée par les lignes qui joignent les prototypes des clusters de la partition P_1^t aux prototype des clusters correspondants dans la partition P_2^t issu de chaque fenêtre (mois) analysée. Cela ne donne pas des trajectoires parfaitement identifiées car certains clusters partagent à un moment donné le même cluster correspondant. On remarque en effet que seuls quatre clusters sont parfaitement identifiés et stables, les autres subissent des changements au cours du temps.

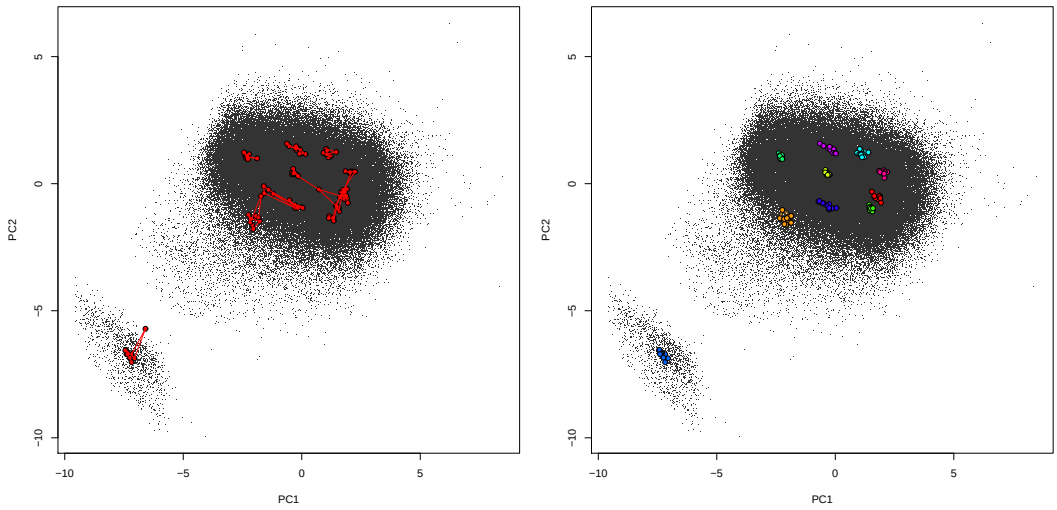

Figure 5.7 – Projection et suivi des prototypes des clusters obtenus par les classifications locales : indépendante (à gauche) et dépendante (à droite).

Nous avons également calculé la variance intra-classe mois par mois pour les

partitions obtenues par les stratégies de *classification globale* CG, *locale indépendante* CL_1 et *locale dépendante* CL_3 (voir figure 5.8). Les partition obtenues par la classification CL_1 sont les plus homogènes car elles présentent les plus petites valeurs d'inertie intra-classe, suivi des partitions obtenues par la classification CL_3 et finalement celles générées par la classification globale CG. Nous pouvons conclure que les clusters obtenus par la stratégie de classification CL_1 présentent plus de cohésion.

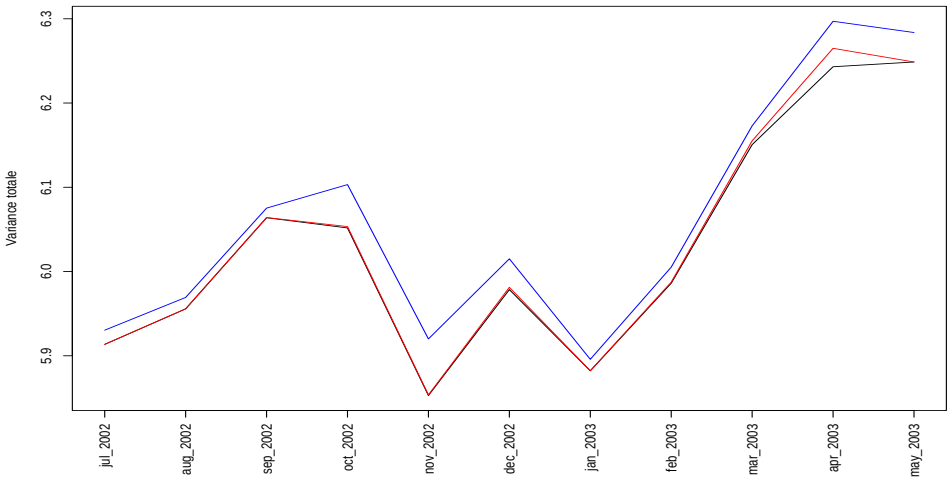

Figure 5.8 – Variance intra-classe des partitions obtenues par les stratégies de classification : locale indépendante CL_1 (trait noir), locale dépendante CL_3 (trait rouge) et globale CG (trait bleu).

La figure 5.9 présente les valeurs obtenues par l'indice corrigé de Rand lors des comparaisons des partitions obtenues sur chaque fenêtre de temps analysée par les différentes stratégies de classification. Les partitions comparées ici sont celles marquées en rouge sur les schémas des stratégies de classification illustrées par les figures 3.2, 3.3, 3.4 et 3.5 du chapitre 3, c'est-a-dire, les partitions finales obtenues par la méthode de classification appliquée sur les individus des fenêtres (mois) analysés. Ces partitions sont confrontées deux-à-deux à l'aide de l'indice de Rand corrigé (CR) et de la F-measure. La figure 5.10 présente sous la forme de boîte à moustache (*boxplot*) les 10 valeurs obtenues (une par cluster) par la F-mesure.

Les valeurs de l'indice CR montrent que les partitions obtenues par la stratégie de classification *locale indépendante* CL_1 sont parfois très différentes de celles obtenues par la stratégie de *classification globale* CG. Ces différences sont également

confirmées par la F-measure. Dans le cas de confrontation entre les stratégies de classification *locale indépendante* versus la stratgégie de classification *globale*, il y a systématiquement des valeurs faibles, c'est-à-dire, certains changements détectés par la classification *locale indépendante* ne sont pas retrouvés par la classification *globale*. On remarque également que la classification *locale précédente* CL_2 ne donne pas de résultats très différents de ceux obtenus par la classification *locale dépendante* CL_3, les clusters obtenus ne changent pas ou peu au cours du temps.

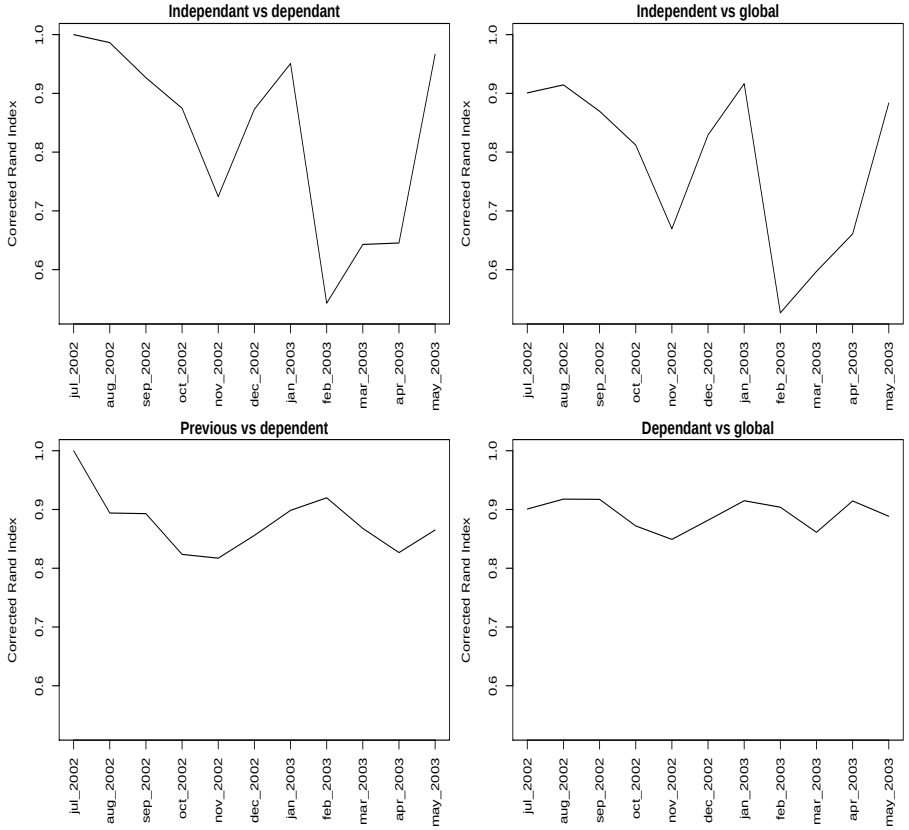

Figure 5.9 – Valeurs de l'indice corrigé de Rand lors des comparaisons des partitions obtenues par les différentes stratégies de classification sur les fenêtres (mois) analysées.

Dans les cas de confrontation entre la classification *globale* versus la classification *locale dépendante* CL_3, ce qui apparaît nettement est le fait que les clusters soient très stables dans le temps. En fait, aucune valeur de la F-mesure n'est au dessous de 0.877, ce qui veut dire que les partitions obtenues par ces deux stratégies de classification sont très semblables. Le même fait est vérifié pour le cas de confrontation de la stratégie de classification *locale précédente* CL_2 versus la stratégie de

classification *locale dépendante* CL_3. En contrepartie, dans le cas de confrontation entre la classification *globale* versus la classification *locale indépendante* CL_1, l'on obtient au contraire des clusters très différents de ceux obtenus globalement. Ce qui est identifié par des valeurs faibles de la F-mesure, inférieures à 0.5 pour certains mois.

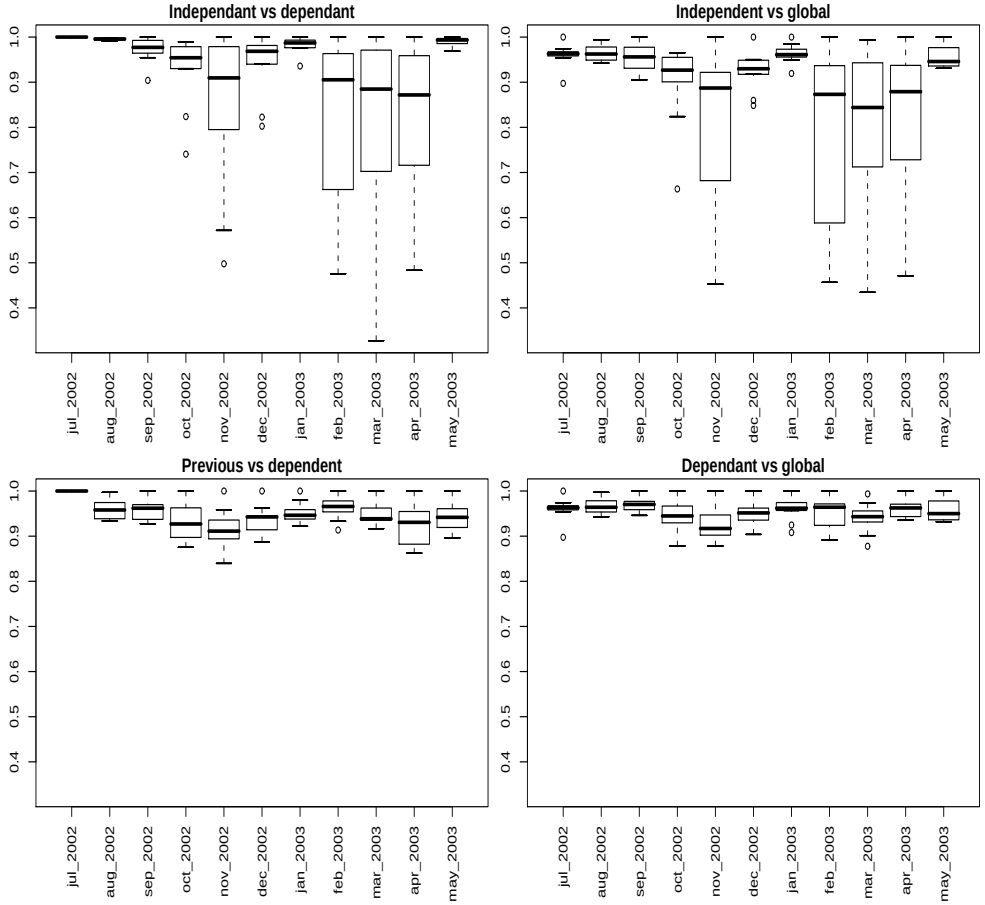

Figure 5.10 – *Boxplots* résumant les valeurs obtenues par la F-measure pour chaque cluster lors de la comparaison des partitions obtenus par les différentes stratégies de classification sur chaque fenêtre (mois) analysée.

Ces résultats viennent confirmer de façon pratique les résultats théoriques démontrés dans la section 4.3 du chapitre 3. En effet, les méthodes MND, MNDS et MNDSO convergent vers les mêmes solutions.

En conclusion, nous pouvons dire que parmi les stratégies de classification locale investiguées, la stratégie de classification ***locale indépendante*** est la plus adaptée à la détection des changements qui peuvent se passer d'une fenêtre de temps à l'autre,

elle est ainsi capable de repérer des changements inaperçus par une classification globale sur tout l'ensemble de données.

Analyse des résultats obtenus par l'approche de détection et suivi des changements appliquée sur le tableau de comptage des données d'usage sur les thèmes de pages du centre d'informatique de l'UFPE

Dans un deuxième temps, nous avons modelisé les données d'usage du centre d'informatique de l'UFPE en tant que tableau de comptage de clics sur différents thèmes de pages (cf. section 2.2 du chapitre 3). Pour ce faire, nous avons extrait de la structure sémantique du site (cf. figure 5.6) le premier niveau de l'hiérarchie des pages, et ainsi défini 10 thèmes de pages, à savoir : thème 1 : *The Informatics' center*, thème 2 : *People*, thème 3 : *Graduation*, thème 4 : *Pos-graduation*, thème 5 : *Extension*, thème 6 : *Research*, thème 7 : *Infrastructure*, thème 8 : *Services*, thème 9 : *News* et thème 10 : *Phones*. Ceci en conservant uniquement les navigations ayant un minimum de 10 clics sur les thèmes décrits, le tableau de comptage final contient un total de 28 220 navigations.

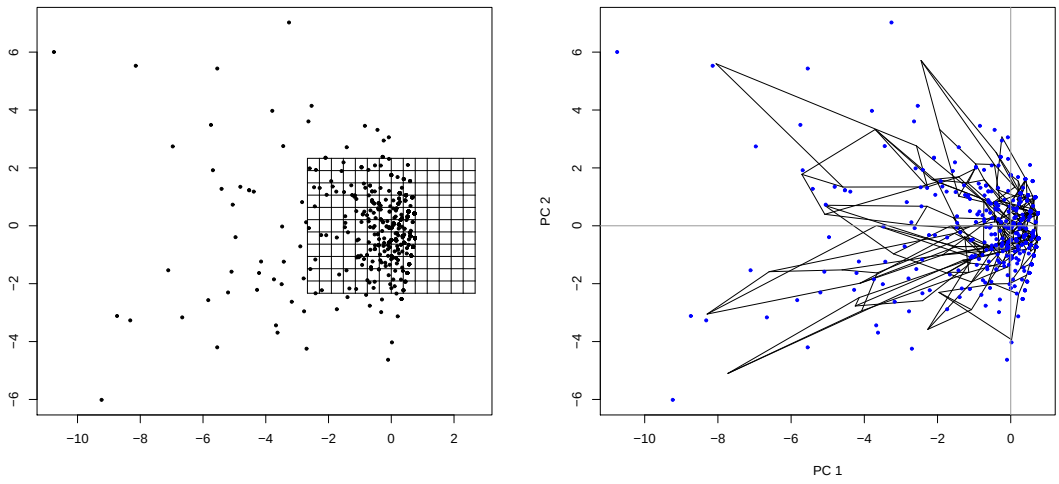

Figure 5.11 – Carte initiale utilisée par la SOM (à gauche) et carte finale obtenue par la SOM sur les individus de la fenêtre 5 (novembre 2002) (à droite).

Pour la classification des individus (navigations) de chaque fenêtre (mois), nous avons appliqué cette fois-ci la méthode des cartes auto-organisatrices de Kohonen

initialisée par une analyse en composantes principales et combinée avec une classification ascendante hiérarchique, utilisant le critère de Ward et la distance Euclidienne, sur les prototypes finaux de la SOM (comme expliqué en début de section). Ceci afin d'obtenir de façon automatique le nombre de clusters sur chaque paquet (fenêtre) de données analysé.

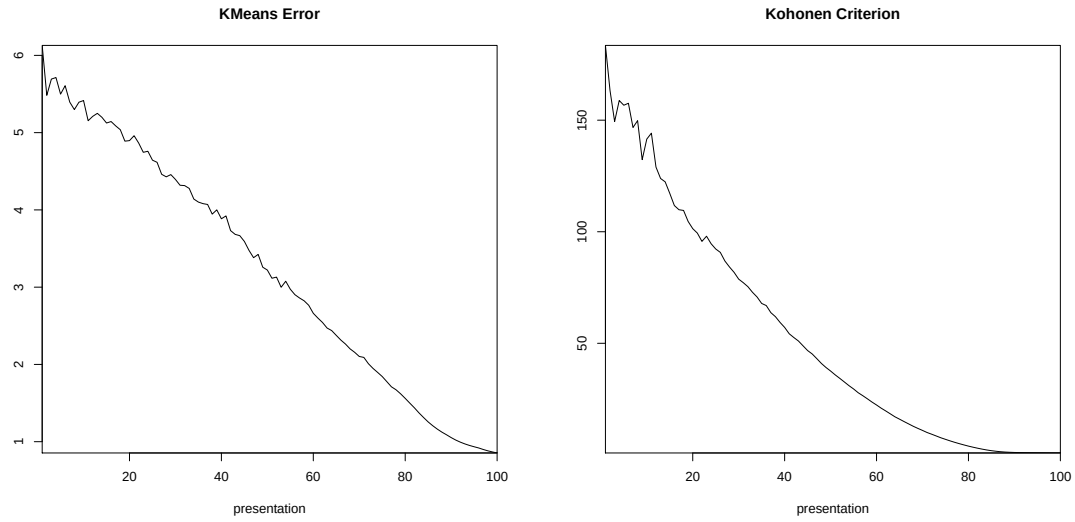

Figure 5.12 – L'erreur k-means (à gauche) et l'erreur kohonen après l'execution de la SOM sur les individus de la fenêtre 5 (novembre 2002) (à droite).

A titre illustratif, la figure 5.11 présente la carte initiale obtenue par l'ACP et la carte finale obtenue par la SOM sur la fenêtre 5 (mois de novembre 2002) contenant un total de 868 individus. Comme montré sur le graphique (a) de cette figure, la grille initiale contient 12 lignes et 15 colonnes, ce qui donne un total de 180 neurones. La figure 5.12 présente l'erreur k-means et l'erreur kohonen (cf. équations 2.9 et 2.10 présentés dans la section 3.2 du chapitre 2) obtenue par la SOM sur cette fenêtre.

La figure 5.13 présente le dendrogramme obtenu par la CAH sur les prototypes finaux issus de la SOM ainsi que les paliers des valeurs relatives des inerties intra-classe. La figure 5.14 présente les valeurs des différences premières et secondes de l'inertie intra-classe de la partition finale obtenue par la SOM. Le nombre de classes retenu est représenté par le premier pic du graphique des différences secondes correspondant à la valeur 3. La figure 5.15 représente la projection sur le premier plan factoriel d'une ACP des classes retenues.

Les prototypes des clusters issus de toutes les fenêtres analysées sont projetés

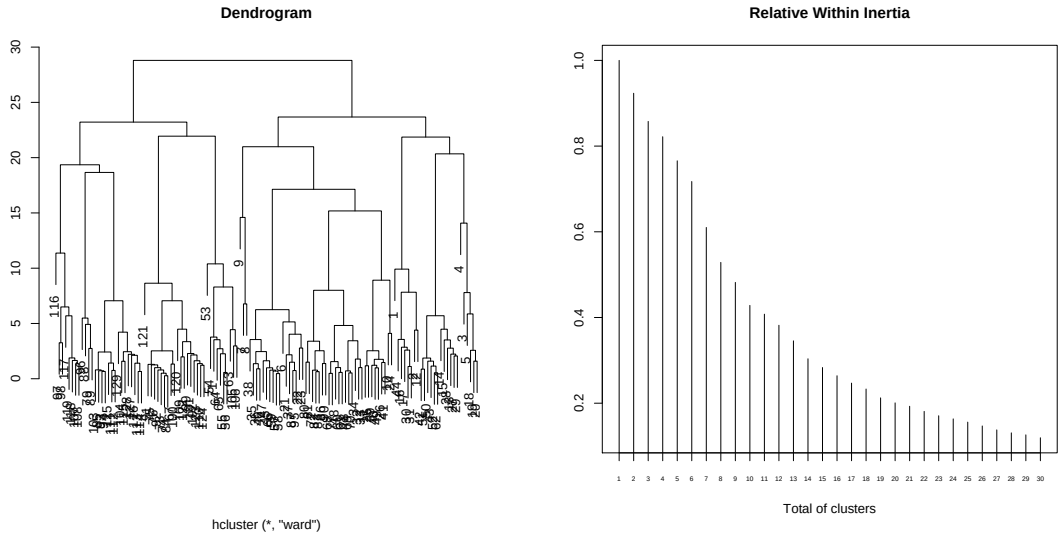

Figure 5.13 – Dendrogramme obtenu par la CAH sur les prototypes finaux issus de la SOM (à gauche) et les valeurs relatives des inerties intra-classe (à droite).

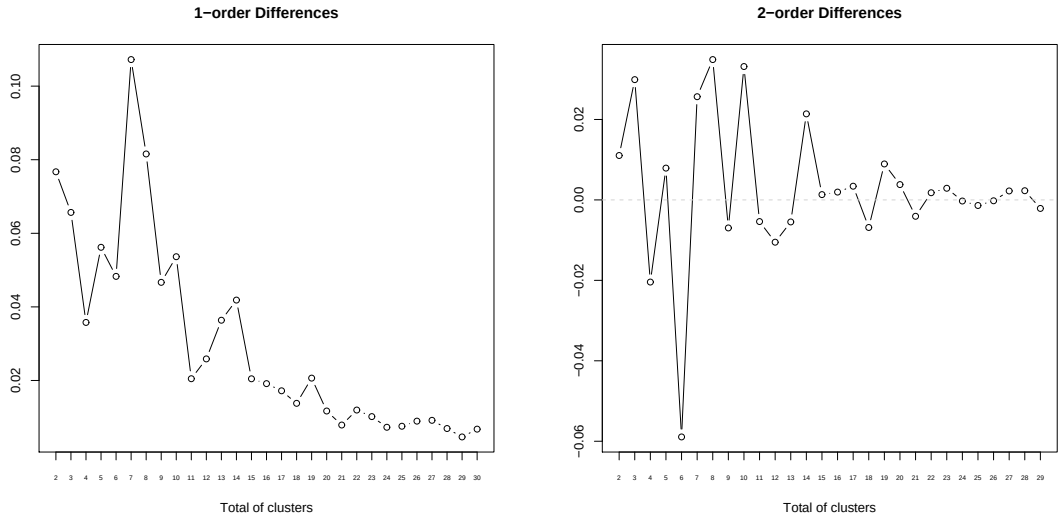

Figure 5.14 – Différences des inerties intra-classe des partitions obtenues par la CAH appliquée sur les prototypes finaux fournis par la SOM sur les individus de la fenêtre 5 (novembre 2002) : premier ordre (à gauche) et second ordre (à droite).

sur le premier plan factoriel d'une ACP illustrés sur la figure 5.16. Sur cette image, les labels du type *(w :i,p :j)* sont reliés au prototype du j^{eme} cluster de la partition obtenue sur la i^{eme} fenêtre. Les traits relient chaque prototype de la fenêtre courante

au prototype du cluster correspondant dans la fenêtre de temps immédiatement suivante. Comme nous pouvons remarquer sur ce graphique, quelques prototypes sont très éloignés des prototypes représentatifs des comportements majoritaires. Par exemple, considerons le prototype labélisé *(w :7,p :2)* localisé à l'extrême gauche de ce graphique. Ce prototype existe uniquement dans la fenêtre 7 (janvier 2003) et ne possède pas de prototype correspondant ni dans la fenêtre précédente (décembre 2002) et ni dans la fenêtre suivante (févier 2003). Ce prototype peut représenter un cluster de comportement transitoire existant uniquement pendant une courte période de temps, dans notre exemple, le mois de janvier 2003. Comme exemple de clusters de profils transitoires, nous pouvons encore citer les clusters liés aux prototypes *(w :3,p :5)* et *(w :4,p :1)* existants uniquement pendant les fenêtres 3 (septembre 2002) et 4 (octobre 2002), ou les prototypes *(w :1,p :3)* et *(w :2,p :1)* des fenêtres 1 (Juillet 2002) et 2 (Août 2002). Tous ces prototypes occupent des positions excentrées sur le grahique de la figure 5.16.

Le graphique de la figure 5.17 exhibe un aperçu du mouvement migratoire des individus appartenant aux différents clusters découverts dans des fenêtres analysées. Sur ce graphique, l'axe des abscisses représente la date de début des fenêtres temporelles, ce qui correspond à la date du premier clic effectué durant la période temporelle concernée. L'axe des ordonnées présente le label des clusters découverts. Les traits pleins lient un cluster spécifique à son cluster correspondant dans la fenêtre de temps immédiatement suivante, celui-ci correspond au cluster ayant attiré la majorité d'individus du cluster de départ. Les traits pointillés illustrent le mouvement migratoire des autres individus appartenant au cluster en question (comme expliqué dans la section 5.2 du chapitre 3).

A partir d'un premier regard sur ce graphique, l'on imagine aisément les difficultés rencontrées par les analystes à la recherche de l'information significative leur permettant de tirer profit de ces résultats. Le dynamisme intrinsèque aux données ainsi que l'occurrence de divers types de changements en simultanée ajoutent une difficulté supplémentaire à l'interprétation de ces résultats.

Une façon de décortiquer cette information implémentée dans notre approche concerne la discrimination sémantique des changements repérés. Pour ce faire, nous appliquons l'algorithme d'interprétation sémantique des changements (présenté dans la section 5.2 du chapitre 3) à partir duquel il est possible de discriminer le type et nombre de changement subis par tous les clusters d'une fenêtre à l'autre. En possession de ces informations détaillées, il est possible de tracer des graphiques

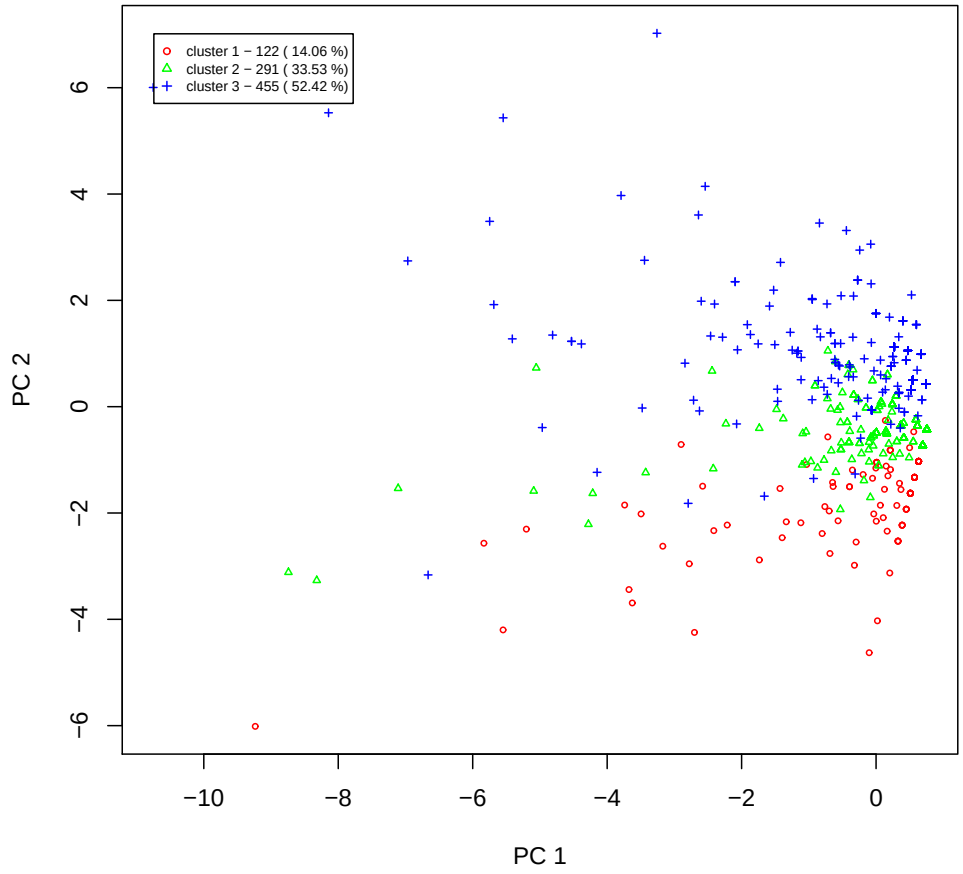

Figure 5.15 – Projection des individus des classes automatiquement repérées sur la fenêtre 5 (novembre 2002).

permettant une interprétation plus claire des changements repères sur le jeu de données analysé. Les graphiques de discrimination sémantique des changements repérés sont présentés sur la figure 5.18.

Il est important de remarquer que la détection de changements se passe entre la fenêtre courante et la fenêtre immédiatement précédente, la première comparaison étant celle entre la deuxième et la première fenêtre. Pour cette raison, dans les graphiques de la figure 5.18, la première fenêtre représentée sur l'axe des abscisses correspond à la deuxième fenêtre analysée et le nombre total de points sur l'axe des abscisses est égal au nombre total de fenêtres - 1, exception faite au graphique

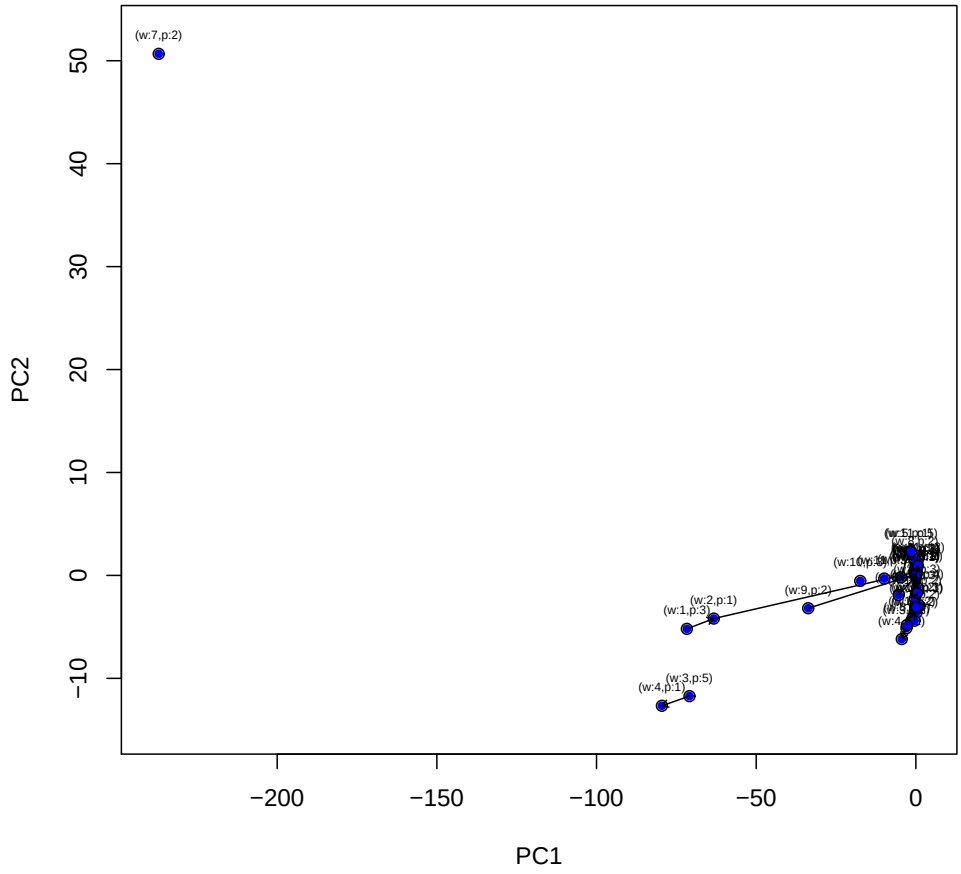

Figure 5.16 – Projection et suivi des prototypes des clusters issus de l'analyse des données d'usage du site Web du CIn-UFPE.

'Total of clusters' où toutes les fenêtres sont présentées sur l'axe des x.

Les valeurs obtenues par l'indice corrigé de Rand ainsi que par la F-measure sur chaque fenêtre (mois) analysée sont présentées sur la figure 5.19. La F-mesure effectue une analyse cluster par cluster en cherchant la meilleure représentation d'un cluster dans la première partition par un cluster correspondant dans la deuxième partition. La F-mesure obtient donc autant de valeurs qu'il y a de clusters dans la première partition. On trace une *boxplot* à partir de ces valeurs pour chaque mois analysé. Les valeurs de la F-measure, détaillées par cluster, sont présentées sur la figure 5.20.

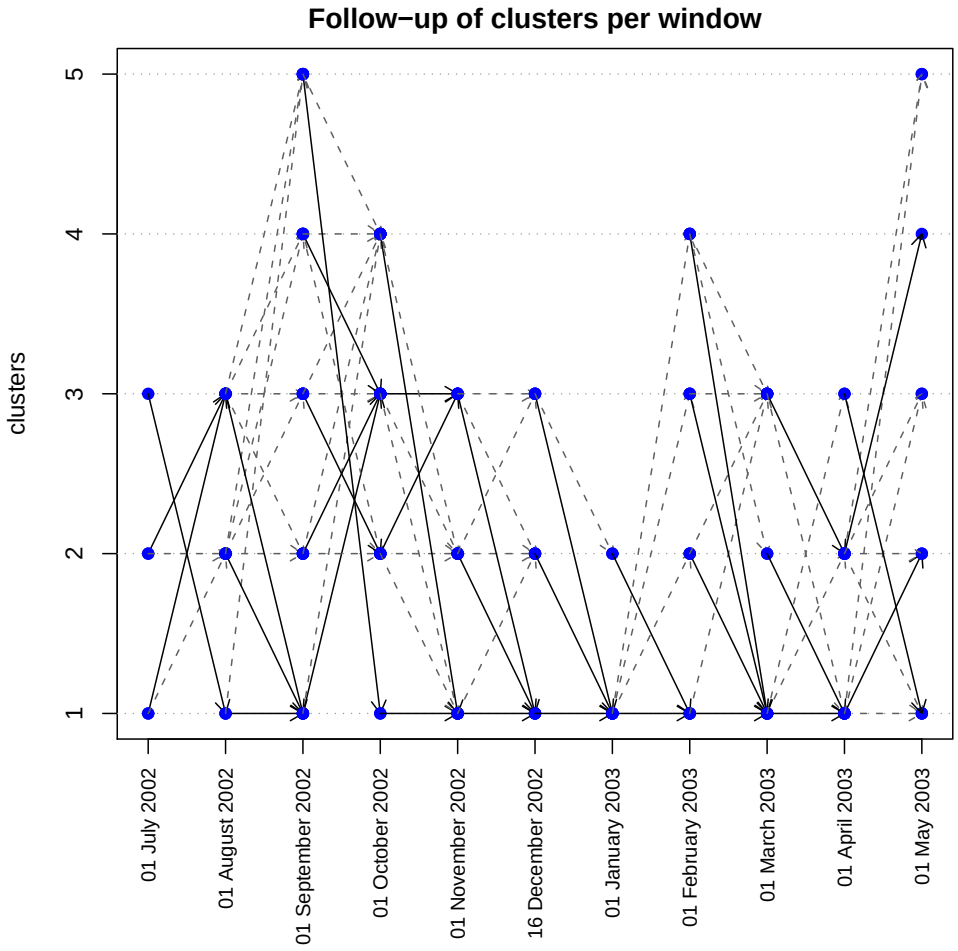

Figure 5.17 – Aperçu du mouvement migratoire des individus entre deux fenêtres consécutives lors de l'analyse des données d'usage du site Web du CIn-UFPE.

Considérons la comparaison faite entre les partitions obtenues sur les individus (navigations) du mois de Novembre 2002. Cette comparaison est effectué entre la partition obtenue par l'affectation des navigations du mois de Novembre 2002 aux prototypes issus de la classification sur le mois de Octobre 2002, et la partition finale obtenue par la méthode de classification appliquée sur les navigations du mois de Novembre 2002. Les valeurs obtenues par les indices de comparaison de partitions sont présentées en quatrième position sur l'axe des abscisses des graphiques de la figure 5.19. La valeur obtenue par l'indice de Rand est de 0.003 et celle obtenue par la F-measure globale est de 0.62. Les valeurs des F-mesures par cluster sont

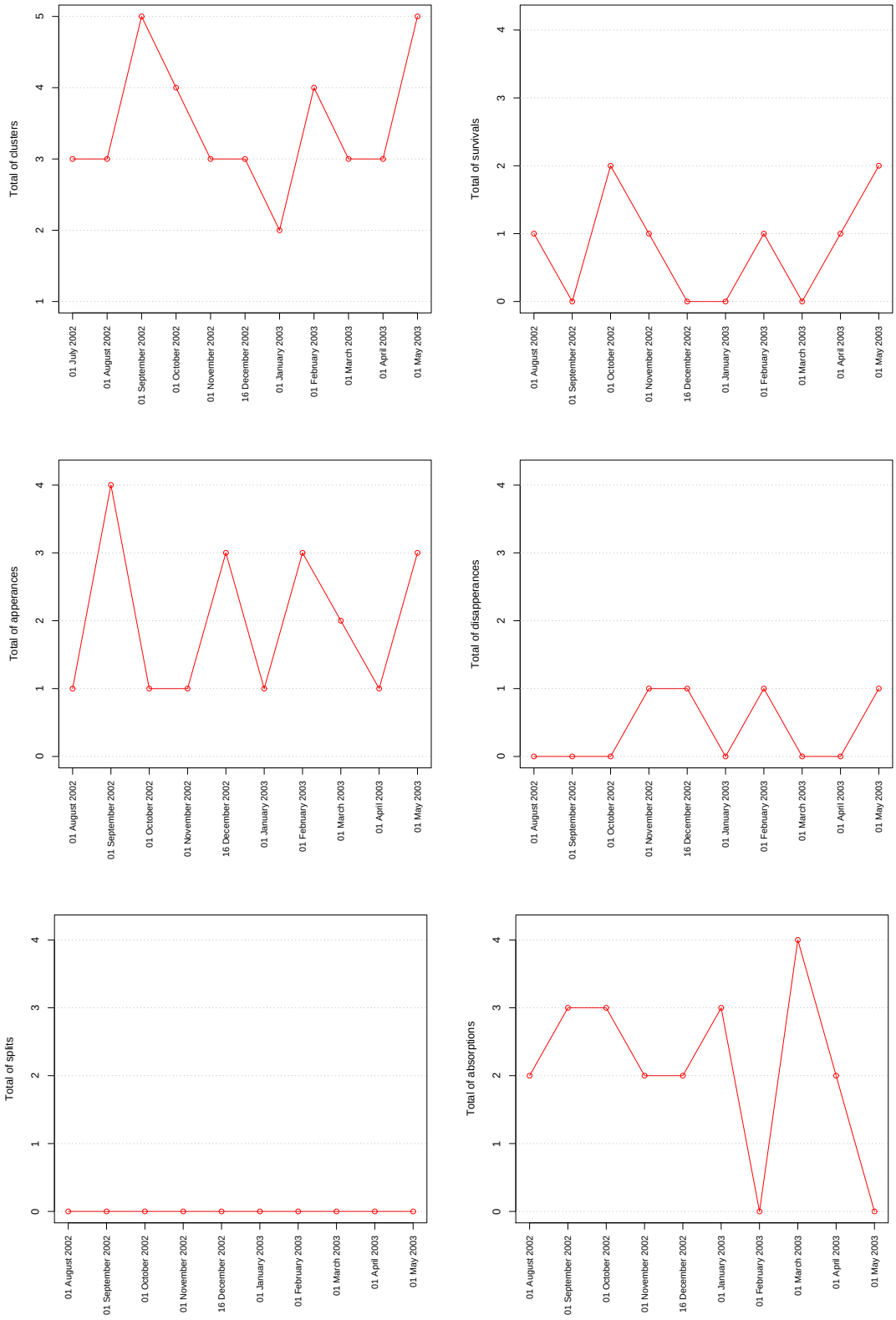

Figure 5.18 – Résumé des changements subis par les clusters lors de l'analyse des données d'usage issues du site Web du CIn-UFPE.

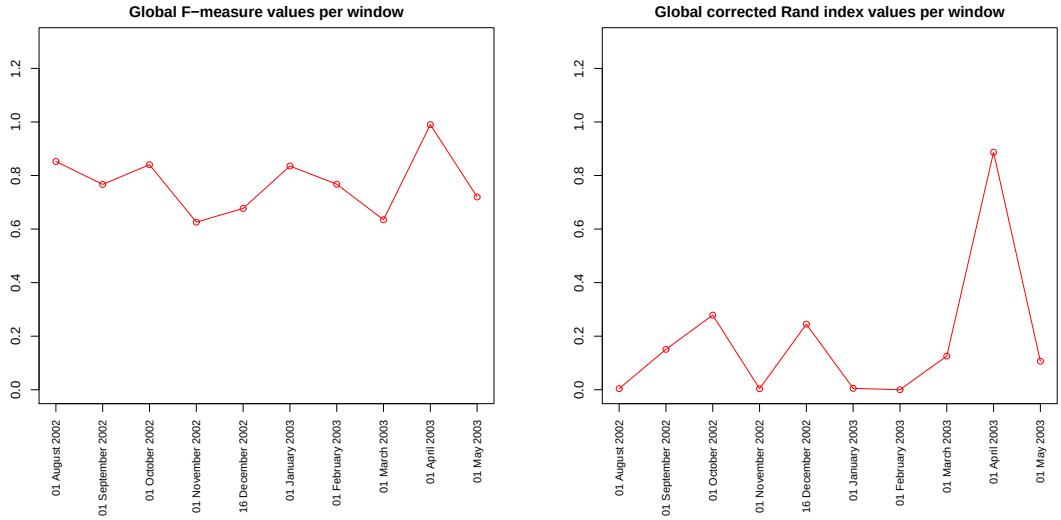

Figure 5.19 – Valeurs de la F-mesure (à gauche) et de l'indice corrigé de Rand (à droite) pour chaque mois lors de l'analyse des données d'usage du site Web du CIn-UFPE.

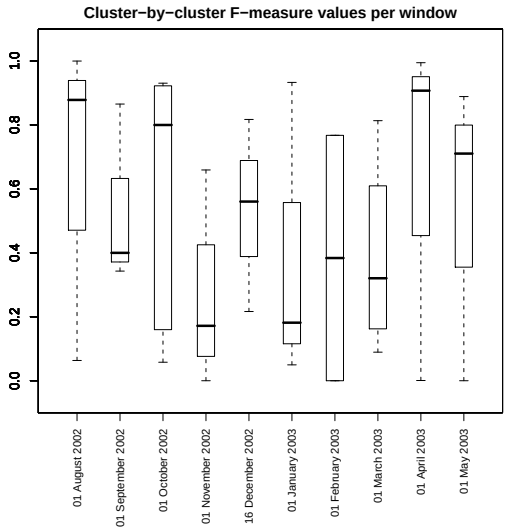

Figure 5.20 – *Boxplots* des valeurs de la F-measure calculée cluster-par-cluster sur chaque mois lors de l'analyse des données d'usage du site Web du CIn-UFPE.

également faibles pour le mois en question (cf. quatrième position sur l'axe des abscisses du graphique de la figure 5.20).

Pour comprendre ce qui s'est passé entre ces deux période de temps, on peut

analyser en détail les graphiques de la figure 5.18. Sur ces graphiques, nous remarquerons que le nombre total de cluster a passé de 4 en *Octobre 2002* à 3 en *Novembre 2002* (cf. graphique *Total of clusters*). L'explication détaillée des changements repérés est la suivante : 1 cluster a survécu (cf. graphique *Total of survivals*), 1 cluster est apparu (cf. graphique *Total of appearances*), 1 cluster a disparu (cf. graphique *Total of disappearances*) et 2 clusters se sont fusionnés (cf. graphique *Total of absorptions*). Nous pouvons également remarquer que sur ce jeu de données, aucune scission n'a été occasionnée (cf. graphique *Total of splits*).

3.2 Étude de cas V : analyse d'un site Web de tourisme

Dans cette étude de cas, nous analysons un jeu de données qui enregistre les traces d'usage du site Web de l'office de tourisme du département de la Nièvre en France [2]. Les données d'usage analysées ici ont été fournies dans le cadre du projet RNTL-ANR Eiffel [3]. Le site en question a pour but de promouvoir le tourisme dans cette région tout en mettant à disposition des internautes des informations pouvant susciter l'intérêt des personnes intéressées par le tourisme dans la région de la Nièvre (suggestions de promenades, d'hébergements, de restauration, etc.). La figure 5.21 présente un aperçu de la page de garde du site Web en question.

Sur la figure 5.22, nous avons un exemple d'information fournie par le site à propos d'un hébergement du type gîte, telle information est conceptualisée par le site comme un objet touristique ayant d'autres informations attachées, telles que le nom et l'adresse de l'hébergement, le prix, la capacité, les services fournis, etc. Ces informations sont facilement récupérables sur un document XML associé à chaque objet touristique, et ce grâce à l'utilisation d'une ontologie du domaine.

Les données d'usage récupérées enregistrent l'accès au site pendant la période du 05/12/2007 à 00 :53 :15 jusqu'au 10/05/2008 à 23 :27 :46. Pour l'analyse de ces données, nous avons appliqué la méthode des cartes auto-organisatrices de Kohonen initialisée par une ACP et combinée avec une CAH sur les données contenues à l'interieur d'une fenêtre temporelle de taille égale à 1 semaine.

2. Le site Web en question est accessible à l'adresse suivante : `http://www.nievre-tourisme.com/`
3. Plus d'informations à l'adresse : `http://www.projet-eiffel.org`

Figure 5.21 – Aperçu de la page de garde du site Web de tourisme du département de la Nièvre.

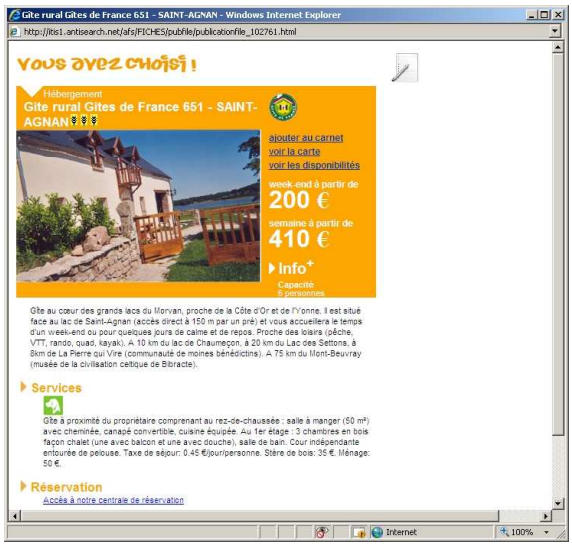

Figure 5.22 – Exemple d'un objet touristique présenté sur le site Web de tourisme de la Nièvre.

Analyse des résultats obtenus par l'approche de détection et suivi des changements appliquée sur le tableau de comptage de l'usage des options de navigation du site Web de tourisme du département de la Nièvre

Un certain nombre d'options de recherche sont mises à disposition des internautes pour faciliter leur recherche d'information sur le site. Par exemple, l'inter-

naute dispose d'un moteur de recherche par mot clé, d'un plan dynamique pour la localisation géographique de l'objet touristique, d'un filtre pour le choix de la langue parlée dans l'établissement, etc. Ces options de recherche sont transmises au serveur Web par intermède de quelques paramètres insérés dans l'URL et peuvent ainsi être facilement repérées dans le champ URL de l'entrepôt de données issu du prétraitement des données d'usage. Autrement dit, les internautes ont la possibilité de rechercher un objet touristique soit en tapant directement leurs requêtes au travers d'un champ textuel, soit en cliquant sur les liens présentés sous un menu. Ces actions engendrent l'exécution du moteur de recherche avec des paramètres spécifiés par l'utilisateur. Le comportement de l'utilisateur peut donc être inféré par l'analyse des paramètres de ce moteur de recherche. Concrêtement, lorsqu'un utilisateur lance une recherche d'objet touristique, un programme nomé *findAll* est exécuté. Nous avons séléctionné 6 paramètres (le programme en question admet plus d'une trentaire de paramètres) qui permettent de nous renseigner sur les différents choix de l'utilisateur, à savoir :

1. **Kewords** : la recherche textuelle
2. **Objetfilter** : le type d'objet touristique (hôtel, restaurant, évenement culturel, etc.)
3. **Geofilter** : la position géographique de l'objet touristique
4. **Labelfilter** : le label de l'établissement
5. **Langfilter** : la langue parlée dans l'établissement
6. **Resfilter** : l'objet touristique en question permet une réservation en ligne

Pour analyser l'utilisation de ces paramètres lors des visistes au site, nous avons défini un tableau de comptage (cf. tableau 5.4). Ce tableau contient une navigation sur chaque ligne et sur les colonnes, les paramètres utilisés par un internaute lors d'une requête sur le site. Chaque cellule de ce tableau contient un chiffre indiquant le nombre de fois que l'internaute a eu recours à un paramètre spécifique durant sa navigation. Les entrées de ce tableau sont ordonnées par date de début de navigation. Pour la période analysée, le tableau de comptage généré continent un total de 29 724 navigations.

Les valeurs obtenues par l'indice corrigé de Rand ainsi que par la F-measure pour la comparaison des partitions sur chaque semaine analysée sont présentées sur la figure 5.23. Les valeurs de la F-measure détaillées par cluster sont exhibées sur fa figure 5.24.

navigation	keywords	objetfilter	geofilter	labelfilter	langfilter	resfilter	date-heure
1	0	0	0	2	2	0	05/12/2007 00 :52 :58
2	0	0	0	1	5	0	05/12/2007 00 :53 :44
3	0	1	0	0	1	0	05/12/2007 01 :53 :46
4	1	1	0	1	1	0	05/12/2007 01 :59 :35
5	0	0	1	0	0	0	05/12/2007 04 :00 :54
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 5.4 – Tableau de comptage des données d'usage du site Web de tourisme de la Nièvre.

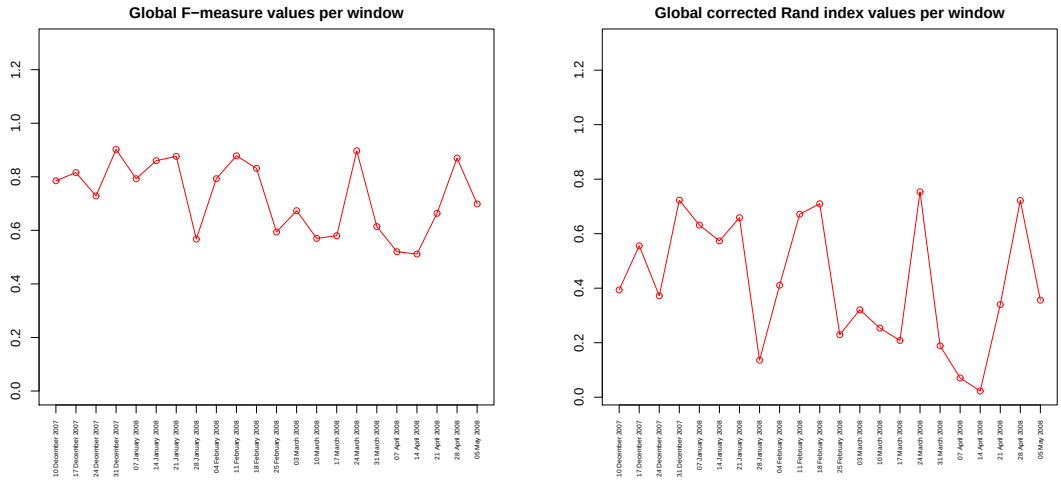

Figure 5.23 – Valeurs obtenues par la F-mesure (à gauche) et par l'indice corrigé de Rand (à droite) pour chaque fenêtre lors de l'analyse de l'usage du site Web de l'office de tourisme de la Nièvre.

Considérons à titre explicatif la comparaison entre les partitions issues de la semaine du 28 janvier 2008 versus celle issue de l'affectation des individus de la semaine courante aux prototypes issus de la semaine précédente (semaine du 21 janvier 2008). La valeur obtenue par les critères de comparaison de partition sont présentées en huitième position sur l'axe des abscisses des graphiques des figures 5.23 et 5.24. La valeur obtenue par l'indice de Rand est de 0.13 et par la F-measure globale est de 0.56. Les valeurs des F-mesures par cluster sont également faibles (cf. huitième position sur l'axe des abscisses du graphique de la figure figure 5.24).

En analysant les graphiques de la figure 5.25 contenant le résumé des changements subis par les clusters entre les fenêtres analysées, on peut analyser plus en

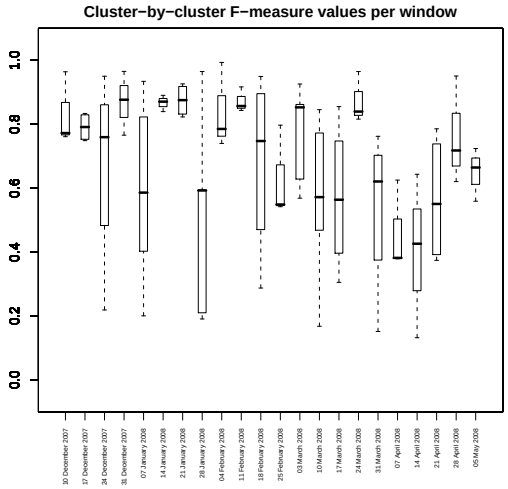

Figure 5.24 – *Boxplots* correspondant aux valeurs obtenues par la F-measure pour chaque cluster sur chaque fenêtre lors de l'analyse de l'usage du site Web de l'office de tourisme de la Nièvre.

détails les changements qui ont eu lieu pendant ces deux semaines. Nous remarquerons que le nombre total de cluster a passé de 5 dans la semaine du 21 janvier 2008 à 3 dans la semaine du 28 janvier 2008 (cf. graphique *Total of clusters*). L'explication détaillée pour ces changements est la suivante : 1 cluster a survécu (cf. graphique *Total of survivals*), 1 cluster s'est scissionné (cf. graphique *Total of splits*) et 3 clusters se sont fusionnés (cf. graphique *Total of absorptions*).

Pour pouvoir décrire les clusters de comportement en fonction de ses variables les plus représentatives, nous cherchons les variables dont les rapports $\frac{B_l^j}{T^j}$ et $\frac{B_l^j}{B_l}$ sont élevés (variables caractérisant le plus chaque cluster), comme décrit dans la section 6 du chapitre 2. Considérons le tableau 5.5 contenant les variables les plus significatives des clusters issus des semaines en question. Nous appliquons la contrainte $\frac{B_l^j}{B_l} > 1/6$ afin d'extraire uniquement les variables ayant un pouvoir discriminant au dessus de la moyenne des variables. Les clusters en question, ainsi que les changements subis par ceux-ci peuvent être interprétés comme suit :

– **Cluster 1** de la semaine du *21 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent uniquement un filtre spécifiant le type d'objet recherché. Ce groupe de comportement s'est scissionné générant les groupes de comportement 1 et 2 de la fenêtre suivante.

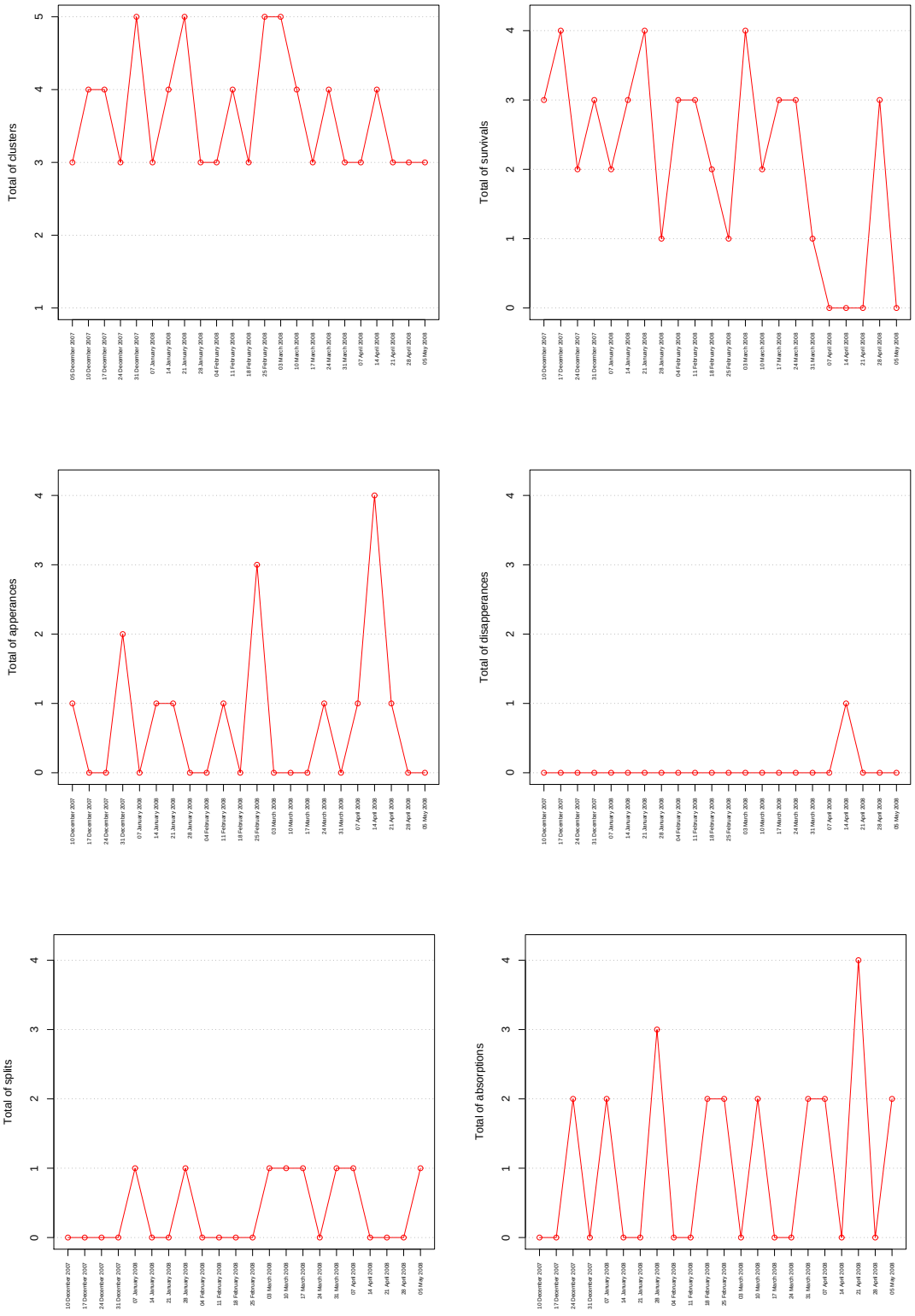

Figure 5.25 – Résumé des changements subis par les clusters lors de l'analyse de l'usage du site Web de l'office de tourisme de la Nièvre.

window	id cluster	variable	$\frac{B_l^j}{T^j}$	$\frac{B_l^j}{B_l}$	$\frac{B^j}{T^j}$	$> \frac{B}{T}$
21 janvier 2008	1	objetfilter	0.0003	0.715	0.0005	oui
21 janvier 2008	2	keywords	0.0002	0.458	0.0003	non
21 janvier 2008	2	objetfilter	0.0001	0.319	0.0005	oui
21 janvier 2008	3	labelfilter	0.0005	0.900	0.0006	oui
21 janvier 2008	4	langfilter	0.0005	0.918	0.0006	oui
21 janvier 2008	5	resfilter	0.0005	0.866	0.0005	oui
28 janvier 2008	1	labelfilter	1.69e-005	0.264	2.53e-005	non
28 janvier 2008	1	langfilter	1.59e-005	0.247	2.39e-005	non
28 janvier 2008	1	resfilter	1.36e-005	0.212	0.0005	oui
28 janvier 2008	1	keywords	1.24e-005	0.193	2.70e-005	non
28 janvier 2008	2	resfilter	2.63e-005	0.580	0.0005	oui
28 janvier 2008	2	labelfilter	8.27e-006	0.182	2.53e-005	non
28 janvier 2008	2	langfilter	8.04e-006	0.177	2.39e-005	non
28 janvier 2008	3	resfilter	0.0004	0.947	0.0005	oui

Table 5.5 – Variables les plus significatives des clusters de comportement issus des semaines du 21 et 28 janvier 2008.

- **Cluster 2** de la semaine du *21 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent simultanément la recherche par mot clé et le filtre spécifiant le type d'objet recherché. Ce groupe de comportement s'est fusionné dans le groupe de comportement 2 de la fenêtre suivante.
- **Cluster 3** de la semaine du *21 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent uniquement l'option de recherche spécifiant le label de l'établissement recherché. Ce groupe de comportement s'est fusionné dans le groupe de comportement 2 de la fenêtre suivante.
- **Cluster 4** de la semaine du *21 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent uniquement l'option de recherche spécifiant la langue parlée dans l'établissement recherché. Ce groupe de comportement s'est fusionné dans le groupe de comportement 2 de la fenêtre suivante.
- **Cluster 5** de la semaine du *21 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent uniquement l'option de recherche spécifiant que l'établissement recherché admet une réservation en ligne. Ce groupe de comportement a survécu et correspond au groupe de comportement 3 de la fenêtre suivante.
- **Cluster 1** de la semaine du *28 janvier 2008* : Groupe de comportement concernant les internautes utilisant simultanément les options de recherche spécifiant le label, la langue parlée, le fait que l'établissement admet une ré-

servation en ligne et encore la recherche par mot clé, dans cet ordre d'importance.

- **Cluster 2** de la semaine du *28 janvier 2008* : Groupe de comportement concernant les internautes utilisant simultanément les options de recherche spécifiant que l'établissement admet une réservation en ligne, ainsi que le label et la langue parlée dans l'établissement recherché.
- **Cluster 3** de la semaine du *28 janvier 2008* : Groupe de comportement concernant les internautes qui utilisent uniquement l'option de recherche spécifiant que l'établissement recherché admet une réservation en ligne.

3.3 Étude de cas VI : analyse d'un jeu de données issu du *marketing*

Le jeu de données analysé dans cette étude de cas concernant le suivi des achats d'un panel de consommateurs mis à disposition dans le cadre du concours ouvert aux jeunes chercheurs du Symposium Apprentissage et Science des Données (SLDS 2009) [4]. Le jeu de données en question concerne le suivi des achats de 10 068 clients pendant 14 mois (du 09 juillet 2007 jusqu'au 08 septembre 2008) sur 2 marchés de biens de consommation. Chaque marché commercialise 3 marques de produits.

Champ	Signification
ident	identification client
date	premier jour de la semaine de l'achat
marché	marché concerné (marche_1, marche_2)
marque	marque du produit acheté (A, B, C, D, E, F)
valeur	valeur des achats

Table 5.6 – Champs descriptifs du jeu de données en *marketing* du SLDS 2009.

Les données à analyser ont été fournies dans un fichier de 3 745 296 lignes contenant les champs décrits dans le tableau 5.6. Dans ce fichier, tous les croisements *date x marché x marque* ont été présentés, même en cas d'absence d'achat où la valeur était nulle. A partir de ces données, nous avons défini un tableau croisé *client x marque* ordonné selon la date de l'achat (cf. tableau 5.7). Après l'élimination des lignes contenant uniquement des valeurs nulles, ce tableau présente un total de 262 215 individus.

4. Plus d'informations sur ce symposium sont disponibles à l'adresse suivante : `http://www.ceremade.dauphine.fr/SLDS2009`

ident	A	B	C	D	E	F	date
2	0	0	0	0	1.65	0	09/07/2007
6	0	0	0	0	1.76	0	09/07/2007
8	7.7	0	0	0	0	0	09/07/2007
10	3.52	0	0.88	0	0	0	09/07/2007
11	1.87	0	0	0	0	0	09/07/2007
12	0	0	0	1.65	0	0	09/07/2007
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 5.7 – Extrait des premières lignes du tableau croisé généré à partir du jeu de données en *marketing* du SLDS 2009.

Analyse des résultats obtenus par l'approche de détection et suivi des changements appliquée sur des données de trace d'achat de différentes marques de produit

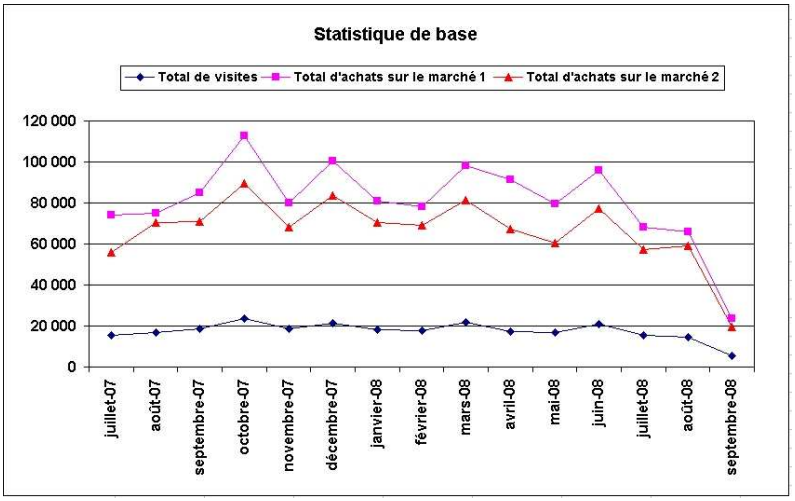

Figure 5.26 – Statistiques de base sur le jeu de données en *marketing* du SLDS 2009.

Durant nos experimentations sur ce jeu de données, nous avons utilisé une fenêtre de temps de taille égale à 1 mois. C'est-à-dire, la méthode de classification non supervisée adoptée est appliquée sur l'ensemble d'individus ayant réalisé des achats dans un même mois, ceci afin de segmenter les clients en fonction de leurs habitudes d'achat dans le temps. La figure 5.26 présente quelques statistiques de base, comme le nombre de visites de clients et le total d'achat par marché et par mois. Comme nous pouvons remarquer, le nombre de visites est assez stable pendant toute la période analysée. Ce nombre avoisine 20 000 visites par semaine. De plus, la valeur

totale d'achats sur le marché 1 est toujours supérieure à celle du marché 2 durant une même période.

Les valeurs obtenues par les deux indices de comparaison de partition (F-measure et indice corrigé de Rand) sont montrées sur la figure 5.27. Les périodes les plus instables sont celles qui présentent les valeurs les plus petites de la F-mesure et de l'indice corrigé de Rand. Sur le jeu de données analysé, ces périodes correspondent aux mois de *Octobre 2007*, *Mars 2008* et *Juillet 2008*. Les valeurs détaillées de la F-measure par cluster sont montrées sur le graphique de la figure 5.28.

Comme nous pouvons remarquer, les changements sont également repérés par l'indice corrigé de Rand en plus grandes ampleurs. Cet indice fournit une mesure globale basée sur tout l'ensemble de clusters dans les deux partitions, et se montre ainsi très sensible aux changements.

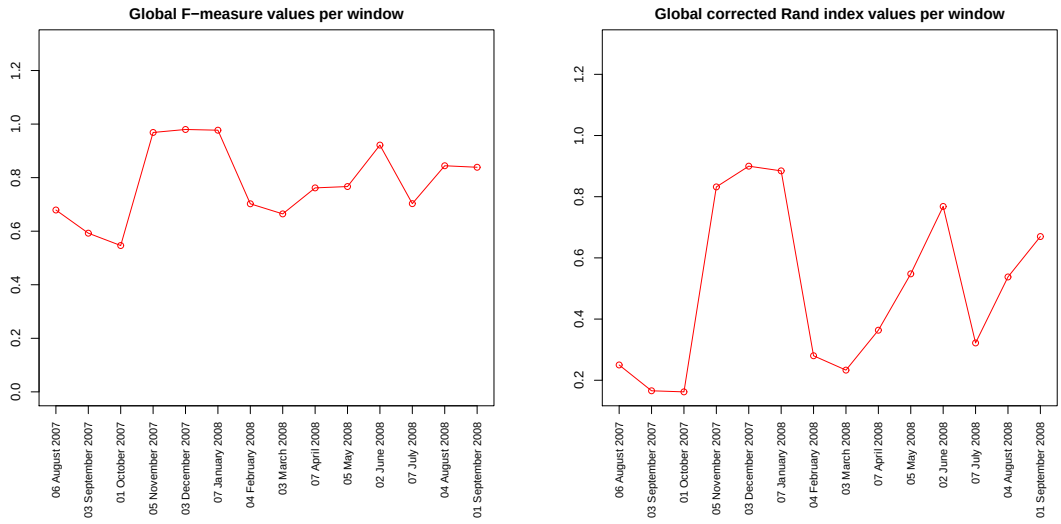

Figure 5.27 – Valeurs obtenues par la F-mesure (à gauche) et par l'indice de Rand corrigé (à droite) lors de l'analyse des données de *marketing* du SLDS 2009.

Les changements subis par les clusters de comportement d'achat tout au long de la période analysée sont résumés dans la figure 5.29. Le nombre total de clusters de préférences d'achat découverts par notre méthode varie entre 3 et 5 (cf. graphique *Total clusters* de la figure 5.29). Un cluster représente un ensemble de clients ayant des préférences d'achat similaires durant un même mois. Remarquons que durant les mois d'*Octobre 2007* jusqu'à *Janvier 2008*, le nombre total de clusters de préférences d'achat a été stabilisé et égal à 3. Ceci peut être une réponse à une stratégie de

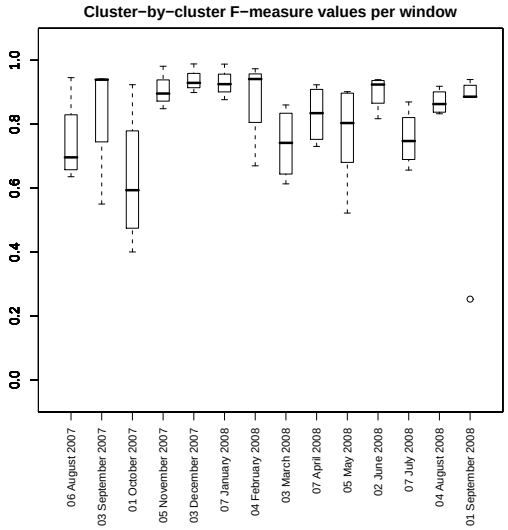

Figure 5.28 – *Boxplots* correspondant aux valeurs obtenues par la F-measure lors de l'analyse des données de *marketing* du SLDS 2009

vente mise en ligne durant cette période de temps. De plus, ces mois correspondent à une période de grande stabilité selon montrent les valeurs élevées de l'indice de Rand et de la F-measure (cf. figures 5.27 et 5.28).

Toujours en cohérence avec les résultats obtenus par la F-mesure et par l'indice corrigé de Rand, notre méthode a détecté que la totalité de clusters de comportement d'achat a survécu durant ces mois de stabilité (cf. graphique *Total of survivals* de la figure 5.29).

Afin de mieux connaître les profils les plus typiques de comportement d'achat des clients, nous avons eu recours à l'interprétation des clusters par ses variables les plus discriminantes. Le tableau 5.8 présente les trois premières variables les plus representatives des clusters découverts pendant le mois de *Octobre 2007* .

Ces profils d'achat peuvent être décrits comme suit :

- **Cluster 1** : groupe des clients ayant une préférence d'achat majoritaire pour les marques F et B en premier plan et la marque A en deuxième plan.
- **Cluster 2** : groupe des clients ayant une préférence d'achat pour la marque F en premier plan et les marques A et E en deuxième plan.
- **Cluster 3** : groupe des clients ayant une préférence d'achat pour la marque B en premier plan et les marques D et E en deuxième plan.

Dans le domaine du *marketing*, si l'on souhaite suivre le comportement d'achat

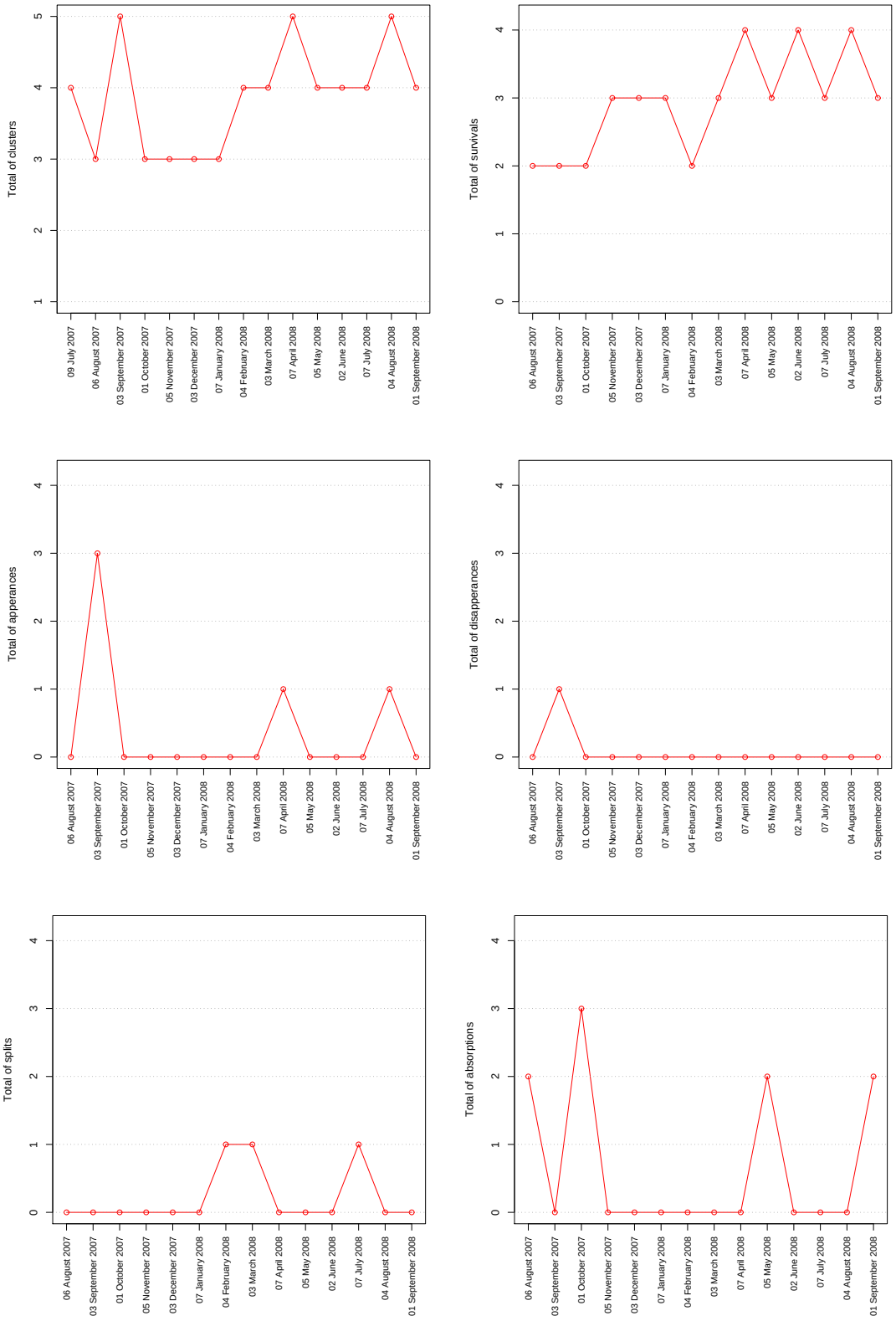

Figure 5.29 – Résumé des changements subis par les clusters issus des données de *marketing* fournies dans le cadre du SLDS 2009.

id cluster	variable	$\frac{B_l^j}{T^j}$	$\frac{B_l^j}{B_l}$	$\frac{B^j}{T^j}$	$> \frac{B}{T}$
1	F	2.23e-006	0.44	3.56e-005	oui
1	B	2.19e-006	0.43	3.11e-005	oui
1	A	2.18e-007	0.04	1.78e-006	non
2	F	3.32e-005	0.93	3.56e-005	oui
2	A	1.31e-006	0.03	1.78e-006	non
2	E	4.24e-007	0.01	1.04e-006	non
3	B	2.88e-005	0.93	3.11e-005	oui
3	D	8.01e-007	0.02	1.38e-006	non
3	E	4.69e-007	0.01	1.04e-006	non

Table 5.8 – Trois premières variables les plus significatives des clusters de comportement issus du mois d'octobre 2007.

d'un client spécifique, il suffit de vérifier si le client en question change de cluster. Si le client reste toujours dans un même cluster au cours du temps, celui-ci peut être considéré *fidèle* aux marques liées aux clusters de préférence d'achat. De l'autre côté, si le client change considérablement de cluster au cours du temps, celui-ci peut être classé tel un *zappeur* entre les marques de produit.

4 Expérimentations sur la détermination du nombre de classes

Dans le domaine de la classification automatique, la détermination du bon nombre de clusters reste un problème ouvert et largement exploité. Malgré la panoplie d'indices proposés dans la littérature, le consensus sur le nombre de clusters à retenir pour un même ensemble de données n'est pas toujours présent parmi les indices de détermination du nombre de clusters. Nous réalisons, dans cette section, une étude exploratoire des indices les plus répandus pour la détermination du bon nombre de clusters, et ce dans un cadre de classification de deux ensembles de données évolutives.

Les indices pour la détermination du bon nombre de clusters comparés dans cette étude sont les suivants (détailles dans la section 4 du chapitre 2) :

- La pseudo-statistique de Calinski et Harabasz (CH)
- L'indice de Baker et Hubert (BH)
- L'indice de Hubert et Levin (HL)
- L'indice Silhouette (S)

– L'indice de Krzanowski et Lai (KL)
– L'indice de Hartigan (H)
– La statistique Gap (G)
– La détermination du nombre de clusters par la coupure du dendrogramme basée sur les dérivées première et seconde (D), détaillée dans la section 4.2 du chapitre 2.

Pour les expérimentations concernant le 8 premiers indices ci-dessus mentionnés, nous avons utilisé les fonctions du package *clusterSim* de **R** [5]. Les indices ici appelés CH, BH et HL sont nommés respectivement G1, G2 et G3 dans ce package.

Nous aborderons, en premier lieu, le cas de classification sur un jeu de données artificielles dont le nombre de classes est connu à l'avance. Pour cette première analyse, nous utilisons le jeu de données contenant cinq classes bien séparées présenté dans la section 2 du présent chapitre. Sur ce jeu de données, nous appliquons l'effet de déplacement des classes vers une classe cible conforme expliqué dans la section 2.3. Cet effet de rapprochement est répété dans 6 fenêtres consécutives et devient plus accentué à chaque nouvelle fenêtre (revoir figure 5.3 pour la visualisation de l'effet de rapprochement des classes).

Le tableau 5.9 discrimine le nombre de classes suggéré par chaque indice pout chaque fenêtre de données analysée. La dernière ligne de ce tableau contient le taux de réussite de chaque indice, c'est-à -dire, le nombre de fois que l'indice concerné a obtenu le vrai nombre de classes.

Fenêtre	CH	BH	HL	KL	DB	H	G	S	D
1	5	5	10	4	5	5	5	5	5
2	5	30	5	4	5	5	8	5	5
3	5	30	6	4	5	5	6	5	5
4	2	30	14	4	30	5	6	4	5
5	2	30	9	4	30	5	5	5	5
6	2	30	9	9	29	5	8	7	5
7	10	30	9	9	30	2	2	10	4
%réussite	42.85	14.28	14.28	0.00	42.85	**85.71**	28.57	57.14	**85.71**

Table 5.9 – Nombre de clusters suggéré par les différents indices pour un jeu de données artificielles.

Comme nous pouvons remarquer, les deux indices les plus performants parmi ceux analysés sont l'indice de Hartigan (H) et l'indice basé sur les dérivées première

5. Ce package est disponible à l'adresse : `http://cran.r-project.org/web/packages/clusterSim`

et seconde (D). Ce dernier a réussi à trouver un total de 4 cluster parmi les 5 existants dans la dernière fenêtre analysée. Cette fenêtre présente une grande difficulté de classification car dans ce cas, les classes présentent un grand niveau de chevauchement, comme montre la figure 5.30 exhibant les clusters obtenus pas le critère des dérivées.

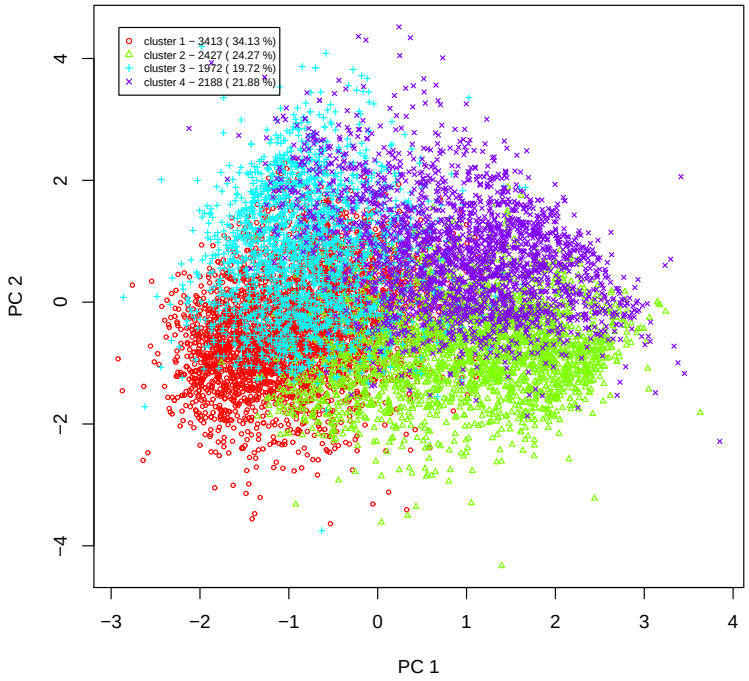

Figure 5.30 – Projection des clusters trouvés par le critère des dérivées pour la fenêtre numéro 7 (facteur de déplacement égal à 0.6).

Pour d'une deuxième analyse, nous utilisons le jeu de données réelles détaillé dans la section 3.1 du présent chapitre. Cette fois-ci, le vrai nombre de clusters dans le jeu de données n'est pas connu à l'avance. Le tableau 5.10 montre le nombre de clusters suggéré par chaque indice pour chaque fenêtre de données analysée.

Prenons, par exemple, le résultat obtenu par le critère des dérivées pour la première fenêtre analysée. La figure 5.31 exhibe les différences premières et secondes des inerties intra-classes relatives des partitions contenant de 2 à 30 classes. Comme nous pouvons remarquer sur le graphique des différences secondes, plusieurs solutions acceptables existent pour le problème de détermination du nombre de clusters

Fenêtre	CH	BH	HL	KL	DB	H	G	S	D
1	30	30	6	2	29	4	3	3	3
2	3	30	2	2	30	5	5	21	3
3	2	2	6	7	2	3	2	2	5
4	2	2	11	3	2	4	2	2	4
5	30	30	12	9	26	2	2	29	3
6	29	30	8	8	30	3	2	30	3
7	2	2	13	24	2	10	3	2	2
8	30	30	8	8	30	2	2	30	4
9	30	25	11	2	30	3	3	25	3
10	30	30	2	4	25	5	3	30	3
11	30	30	7	4	28	2	2	30	5

Table 5.10 – Nombre de clusters suggéré par les différents indices pour le jeu de données d'usage du CIn-UFPE.

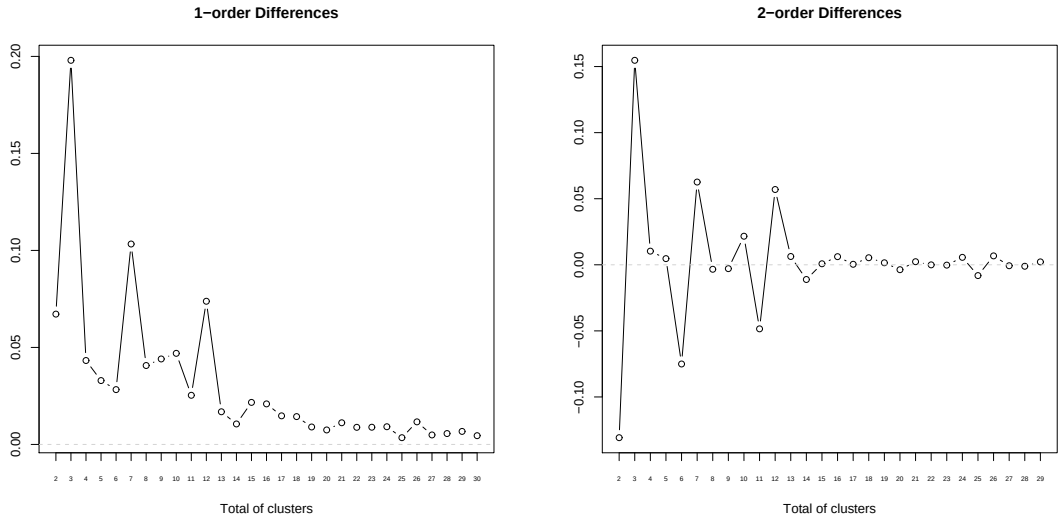

Figure 5.31 – Différences des inerties intra-classe : premier ordre (à gauche) et second ordre (à droite).

contenu dans ce jeu de données. Sur le graphique des différences secondes, nous avons des pics sur les valeurs 3, 7, 10 et 12. Ces solutions sont également trouvées sur les paliers de la figure 5.32 qui montre la valeur relative des inerties intra-classe des partitions obtenues par la CAH sur les prototypes issus d'une classification appliquant la méthode de Kohonen avec une grille initiale de 200 neurones.

Dans notre stratégie, nous retenons la première valeur proposée. Dans l'exemple, la valeur retenue pour le bon nombre de classes est égale à 3. Cette valeur a égale-

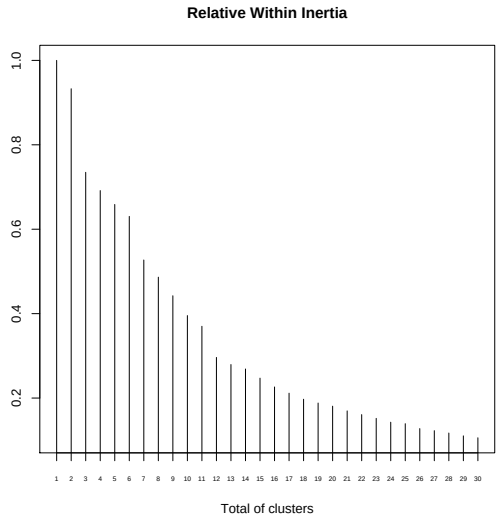

Figure 5.32 – Valeurs relatives des inerties intra-classe.

ment été repérée par la statistique Gap (G) et l'indice Silhouette (S) pour la fenêtre en question. Néanmoins, les indices de détermination du bon nombre de clusters ne sont pas toujours d'accord et le choix du nombre de clusters s'avère une tâche assez délicate.

Après plusieurs analyses réalisées sur différents jeux de données avec les indices cités ci-dessus, nous avons décidé d'adopter le critère basé sur les dérivées première et seconde pour la détermination du bon nombre de clusters durant de nos expérimentations. Si bien que le problème de détermination du bon nombre de clusters ne soit pas concerné par notre approche, nous avons dû choisir un critère à titre expérimental durant de nos expérimentations. Comme évoqué précédemment, notre approche se place dans un plus haut niveau d'abstraction et est totalement indépendante de la méthode de classification ainsi que des critères pour la détermination du nombre de clusters. Ainsi, d'autres critères de détermination du nombre de clusters peuvent sans aucune contrainte être appliqués avec l'approche proposée.

5 Synthèse

Dans ce chapitre, nous montrons, à travers des exemples concrets, l'application de notre approche. Entre autres, nous avons montré que l'analyse des données évolutives par sous-périodes (fenêtres) offre un certain nombre d'avantages, comme par

exemple, cette stratégie rend la méthode de classification plus efficace dans la découverte de nouveaux clusters que l'on marquerait par une analyse globale sur la totalité de données disponibles.

A partir de l'analyse exploratoire sur les stratégies de classification, nous avons montré que les résultats obtenus sont vraisemblablement différents si l'on applique la stratégie de *classification locale indépendante*. Cette stratégie permet notamment de retrouver des clusters non identifiés par les autres stratégies de classification analysées.

Tout au long des expérimentations détaillées dans ce chapitre, nous avons pu remarquer que l'indice corrigé de Rand se montre plus sensible aux différences présentes entre les partitions comparées. La F-measure étant, quant à elle, plus facile à interpréter puisque son analyse est faite cluster par cluster, alors que l'indice corrigé de Rand fournit une mesure globale basée sur tout l'ensemble de clusters dans les partitions comparées.

Enfin, l'interprétation sémantique des changements peut permettre même aux non spécialistes de comprendre l'évolution des données à l'aide de graphiques résumant les changements repérés.

Chapitre 6

Conclusion et perspectives

1 Apports vers la communauté de l'analyse de données

Au cours de ce travail de recherche, nous avons étudié le comportement des utilisateurs des sites Web à partir de l'exploitation des données d'usage disponibles dans les fichiers log gérés par le serveur Web. L'enjeu est ici important à l'heure où la fréquence des visites ainsi que le nombre d'informations mises en ligne augmentent de manière colossale. De plus, les groupes de comportement peuvent être influencés par des facteurs temporels tels que l'heure et le jour de la semaine, des événements saisonniers (ex. vacances d'été, d'hiver, Noël), événements externes dans le monde (ex. épidémies, crises économiques, catastrophes), etc. Les données à analyser deviennent ainsi de plus en plus volumineuses et dynamiques. Malgré cela, les propriétaires des sites Web cherchent à comprendre le comportement de leurs visiteurs afin d'améliorer leur service et ainsi pouvoir répondre de manière précise aux attentes des usagers.

Pour répondre à ces besoins, il est nécessaire de définir des méthodes efficaces permettant le traitement en temps raisonnable d'une telle masse de données. Or, les méthodes de fouille de données les plus traditionnelles considèrent dans leur analyse tout l'ensemble de données disponible. Ceci reste impraticable dans le cas d'un volume de données trop important. De plus, les patterns minoritaires seraient effacés par les plus dominants. Les comportements majoritaires dominent le processus de classification au point d'empêcher l'analyse des comportements plus rares. La caractérisation des comportements d'usage ainsi obtenue se limite alors aux grandes

tendances. Ainsi, il est désirable d'avoir une sous-division de l'ensemble de données à analyser en sous périodes plus significatives avant d'appliquer la méthode de classification, soit en intervalles réguliers et uniformes, soit en intervalles définis de façon adaptée, et ce afin de mettre en évidence l'occurrence de comportements significatifs. Cette course à l'exploitation des données d'usage se solde par un manque de méthodes de fouille de données adaptées au dynamisme du Web et à la dimension temporelle.

Dans le cadre de cette thèse, nous avons proposé une approche de classification ayant pour but de pallier cette problématique. Cette approche s'appuie sur l'analyse des données par sous-périodes de temps, modélisées en tant que fenêtres non recouvrantes de taille fixe, et fait appel à des indices basés sur l'extension de la classification pour la comparaison de partitions issues d'un même ensemble d'individus. Nous avons combiné ces deux stratégies avec une méthode de classification afin de pouvoir repérer les changements sur les données évolutives et tracer un suivi de l'évolution des clusters des profils d'usage. Et ce, tout en effectuant un résumé des données analysées au cours du temps. Cette approche permet ainsi d'appliquer des algorithmes de classification a priori non incrémentaux dans un cadre de traitement incrémental. Après cette étape de définition de l'approche, la principale motivation fut d'y ajouter une fonctionnalité additionnelle permettant l'interprétation sémantique des changements repérés. Ces explorations sont importantes dans la mesure où elles permettent aux concepteurs du site Web d'accompagner et d'interpréter l'évolution des profils de leurs visiteurs ainsi que de repérer les comportements minoritaires.

Si bien que motivée et initialement prévue pour le traitement des données volumineuses et évolutives issues du Web, l'approche proposée est parfaitement adaptée à d'autres contextes où l'analyse de données évolutives s'avère nécessaire.

Les apports de cette thèse concernent encore la modélisation des données évolutives d'usage du Web ainsi que la génération automatique de données artificielles à travers la définition d'une méthodologie permettant la simulation de données d'usage ainsi que l'insertion de changements a priori.

2 Apports vers l'utilisateur final

De nos jours, il semble impossible pour les directeurs d'entreprises de procéder à un traitement manuel de leurs données avant la prise d'une décision importante. En effet, la masse de données est telle qu'un traitement non automatisé de celle-ci semble voué à l'échec. Ceci explique, en partie, le besoin pour ces entreprises de se tourner vers des techniques permettant de fournir des informations pertinentes et dans des délais acceptables.

L'approche que nous avons définie et utilisée tout au long de ce travail a pour objectif de définir une nouvelle stratégie automatisée capable de fournir des moyens nécessaires aux responsables d'un site Web de manière à ce qu'ils puissent être avertis lors de l'évolution des comportements de leurs visiteurs. Il est aussi envisageable que l'administrateur du site puisse mesurer l'impact et l'acceptation d'une nouvelle stratégie de présentation ou de vente d'un produit mise en ligne.

L'analyse des comportements des internautes sur des sites Web est un enjeu important, même en dehors de la sphère marchande. On peut, ainsi, améliorer l'accès à l'information, promouvoir des parties du site faiblement visitées, abandonner des développements sans succès, aider l'internaute pendant sa visite, adapter le site à différents modes de navigation, etc. Il est donc important d'étudier si un site atteint ses buts en confrontant ceux-ci à l'analyse des navigations réalisées par les utilisateurs.

Cela évoqué, supposons maintenant que l'approche proposée dans cette thèse soit mise en place avec un fenêtrage temporel de taille égale à une semaine. L'opérateur du site pourrait ainsi avoir un rapport du changement des profils des visites sur le site entre chaque semaine analysée. Celui-ci pourrait vérifier, par exemple, si les stratégies adoptées pour la présentation de certaines informations ont été efficaces. Cette pratique permettrait par exemple à l'opérateur de confronter sa vision éditoriale du site à la perception de celui-ci par les internautes (telle quelle apparait dans les données d'usage) et de suivre l'évolution du comportement des internautes en fonction des modifications de son site.

Si l'opérateur du site souhaite être averti lors de l'apparition d'un nouveau profil d'usage lié à des thèmes de pages spécifiques, une version *on-line* de notre approche peut être envisageable. Le système pourrait être activé lorsqu'un nouveau paquet (fenêtre) de navigations est enregistré sur le site et ainsi générer une alarme quand le cluster de comportement représentant le profil spécifique reçoit un nouvel individu

ou bien quand son effectif dépasse un seuil préétabli.

3 Perspectives

A l'issue de cette thèse, des perspectives de recherche naissent au fil des pistes abordées au cours de ce travail. Nous pouvons citer la mise en œuvre de techniques permettant de déterminer de façon adaptative la taille de la fenêtre à adopter. Entre autre, nous pouvons signaler la combinaison d'autres indices de validation externe permettant de ressortir les divergences entre deux partitions. À court terme, nous pouvons envisager la codification d'un outil interactif pour la visualisation des résultats obtenus. A long terme, l'élaboration d'une version *online* du système installée du côté serveur peut être envisageable.

Annexe A : Outils d'investigation

L'implémentation est une partie primordiale dans ce travail. Au cours de cette annexe, nous allons décrire brièvement l'ensemble d'outils d'investigation utilisés durant les étapes de codification et d'analyse des résultats.

Le langage *Java* et l'environnement de développement intégré *Eclipse IDE*

Java est un langage orienté objet largement répandu. Ce langage a été crée par Sun Microsystems [1] et présenté officiellement en 1995.

Le langage Java a la particularité principale que les logiciels écrits avec ce langage sont facilement portables sur plusieurs systèmes d'exploitation. La portabilité du code java est assuré par une machine virtuelle [2].

Java permet de développer des applications client-serveur. Côté client, les *applets* sont à l'origine de la notoriété du langage. C'est surtout côté serveur que Java s'est imposé dans le milieu des entreprises grâce aux *servlets* et aux JSP (JavaServer Pages) qui peuvent se substituer à PHP, ASP et ASP.NET.

En Java, une grande partie de problèmes liés à l'allocation de la mémoire est géré par le ramasse-miettes (ou *garbage collector*, en anglais). Cet élément contribue à la robustesse et à la performance des programmes. Le ramasse-miettes est totalement transparent pour le développeur : il est appelé régulièrement et automatiquement pendant l'exécution du programme.

Eclipse IDE [3] est un environnement de développement intégré (*Integrated Deve-*

1. `http://java.sun.com/`
2. Le sens originel de machine virtuelle (ou *Virtual Machine* (VM) en anglais) est la création de plusieurs environnements d'exécution sur un seul ordinateur, dont chacun émule l'ordinateur hôte
3. `http://www.eclipse.org/`

lopment Environment (IDE), en anglais) libre, extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation.

Le langage *R* et l'interface *JRI*

R est un langage de programmation et un environnement mathématique utilisé pour le traitement de données et l'analyse statistique. R est librement disponible sur le réseau CRAN (*Comprehensive R Archive Network*, en anglais) [4]. Ce logiciel a été développé aux laboratoires Bell aux Etats-Unis dans les années 1970 et est diffusé sous licence GNU [5].

Le langage R dispose, entre autres, d'une large collection d'outils statistiques et graphiques ainsi que d'interfaces pour d'autres langues. Le code R est continuellement enrichi grâce aux packages (ou extensions) écrits en R et mis librement à disposition. Des versions compilées de R sont disponibles pour Linux, Windows et Mac OS X.

JRI (Java R Interface) [6] est une bibliothèque JNI (*Java Native Interface*, en anglais) pour communiquer en Java avec R. Fondamentalement, JRI charge la bibliothèque dynamique de R dans Java et permet ainsi l'exécution de codes R à l'intérieur des applications Java. JRI est compatible avec toutes les plates-formes où Java est disponible, y compris Windows, Mac OS et Linux.

Le langage *SQL*, le système de gestion de bases de données *MySQL* et l'API *JDBC*

SQL (*Structured Query Language*, en anglais, ou langage structuré de requêtes, en français) est un langage standard et normalisé, destiné à interroger ou à manipuler une base de données relationnelle. SQL se décompose en 5 parties, à savoir :

– un langage de définition de données (Data Definition Language (DDL), en anglais) : permet de créer et modifier la structure d'une base de données.

4. `http://cran.r-project.org`
5. Projet créé en 1984 pour développer un système d'exploitation complet et libre.
6. `http://www.rforge.net/JRI/`

- un langage de manipulation de données (Data Manipulation Language (DML), en anglais) : permet de sélectionner, insérer, modifier ou supprimer des données dans une table d'une base de données. Celle-ci est la partie la plus répandue de SQL.
- un langage de contrôle de données (Data Control Language (DCL), en anglais) : permet de gérer les privilèges des utilisateurs d'une base de données.
- un langage de contrôle des transactions (Transaction Control Language (TCL), en anglais) : permet le contrôle transactionnel dans une base de données, c'est-à-dire, la validation, l'annulation et contrôle de concurrence des modifications.
- et d'autres modules destinés notamment à écrire des routines (procédures, fonctions ou déclencheurs) et interagir avec des langages externes.

Le modèle relationnel a été crée par E.F. Codd (Directeur de recherche du centre IBM de San José) en 1970, suite à quoi de nombreux langages ont fait leur apparition. MySQL [7] est un système de gestion de bases de données relationnel (Relationnal Database Management System (DBMS) en anglais) fonctionnant sous Linux et Windows. Les principaux avantages de MySQL sont sa rapidité, sa robustesse et sa facilité d'utilisation et d'administration.

JDBC (Java DataBase Connectivity, en anglais) est une API [8] permettant aux applications Java d'accéder à une base de données par le biais d'une interface commune à des sources de données pour lesquelles il existe des pilotes JDBC. Normalement, des pilotes JDBC sont disponibles pour tous les systèmes connus de bases de données relationnelles (par exemple, MySQL).

7. `http://www.mysql.com`

8. Une API (Application Programming Interface, en anglais) est un ensemble de fonctions, procédures ou classes considérées utiles et mises à disposition d'autres programmes.

Annexe B : Publications

Articles de revue

1. **DA SILVA, Alzennyr**. *Diverses approches permettant l'introduction du temps dans la fouille de données d'usage du Web*. Editeurs : Chantal Reynaud et Gilles Venturini. Numéro spécial sur la fouille du Web de la Revue des Nouvelles Technologies de l'Information (RNTI W-1), pages 35-55, ISBN 978.2.85428.793.6, cépaduès éditions, 2007.

2. **DA SILVA, Alzennyr**. *Analyzing the Evolution of Web Usage Data*. In Special issue on Data Stream Analysis of MODULAD (Monde des Utilisateurs de L'Analyse de Données), numéro 36, pages 75-84, May, 2007.

Chapitres de livre

3. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Clustering Dynamic Web Usage Data*. In Nadia Nedjah, Luiza Mourelle and Janusz Kacprzyk (Editors), Springer Berlin/Heidelberg, Series : Studies in Computational Intelligence, Innovative Applications in Data Mining, Vol. 169, pages 71-82, ISBN : 978-3-540-88044-8, 2009.

4. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, DE CARVALHO, Francisco. *Comparing Clustering on Symbolic Data*. In Nadia Nedjah, Luiza Mourelle, Janusz Kacprzyk, Felipe França and Alberto De Souza (Editors), Springer Berlin/Heidelberg, Series : Studies in Computational Intelligence, Intelligent Text Categorization and Clustering, Vol. 164, pages 81-94, ISBN : 978-3-540-85643-6, 2009.

Articles publiés dans des conférences

5. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, DE CARVALHO, Francisco. *Approche pour le suivi des changements sur des données évolutives : application aux données du mar-*

keting. In XVIèmes Rencontres de la Société Francophone de Classification (SFC 2009), Grenoble, 2009.

6. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, DE CARVALHO, Francisco. *Monitoring Data Changes through a Clustering Approach*. In International Federation of Classification Societies (IFCS 2009). Dresden, March 13-18, 2009.

7. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves. *Vers la simulation et la détection des changements des données évolutives d'usage du Web*. In Actes des 9èmes journées Extraction et Gestion des Connaissances (EGC 2009), Revue des Nouvelles Technologies de l'Information (RNTI E-15), pages 453-454, ISBN 978.2.85428.878.0, cépaduès, Strasbourg, January 28-30, 2009.

8. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves. *Stratégies de classification non supervisée sur fenêtres superposées : application aux données d'usage du Web*. In Actes des 8èmes journées Extraction et Gestion des Connaissances (EGC 2008), Revue des Nouvelles Technologies de l'Information (RNTI-E-11), Cépaduès-éditions, pages 219-220, Sophia-Antipolis, France, January 29 - February 1, 2008.

9. **DA SILVA, Alzennyr**, DE CARVALHO, Francisco, LECHEVALLIER, Yves. *Analysing Distance Measures for Symbolic Data Based on Fuzzy Clustering*. In Luiza Mourelle, Nadia Nedjah and Janusz Kacprzyk editors, Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007), pages 109-114, ISBN : 978-0-7695-2976-9, IEEE Computer Society, Rio de Janeiro, Brazil, 22-24 October, 2007.

10. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Construction and Analysis of Evolving Data Summaries : an Application on Web Usage Data*. In Luiza Mourelle, Nadia Nedjah and Janusz Kacprzyk editors, Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007), Pages 377-380, ISBN : 978-0-7695-2976-9, IEEE Computer Society, Rio de Janeiro, Brazil, 22-24 October, 2007.

11. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Clustering Strategies for Detecting Changes on Web Usage Data*. In 56th Session of the International Statistical Institute (ISI 2007), Lisbon, Portugal, 22-29 August, 2007.

12. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Groupement de données évolutives dans la fouille d'usage du Web*. In 39èmes Journées de Statistique de la SFdS, Angers, France, 11-15 June 2007.

13. **DA SILVA, Alzennyr**, DE CARVALHO, Francisco, LECHEVALLIER, Yves, TROUSSE, Brigitte. *Mining Web Usage Data for Discovering Navigation Clusters.* In : XI IEEE Symposium on Computers and Communications (ISCC 2006), pages 910-915, ISBN ĨSSN :1530-1346, 0-7695-2588-1, Mining Web Usage Data for Discovering Navigation Clusters, IEEE Computer Society, Pula-Cagliari, Italy, 2006.

14. **DA SILVA, Alzennyr**, DE CARVALHO, Francisco, LECHEVALLIER, TROUSSE, Brigitte. *Characterizing Visitor Groups from Web Data Streams.* In : IEEE International Conference on Granular Computing (GrC 2006), Atlanta, USA, 2006, pages 389-392, ISBN : 1-4244-0134-8, IEEE Computer Society, 2006.

15. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Construction et analyse de résumés de données évolutives : application aux données d'usage du Web.* In Actes des 7èmes journées Extraction et Gestion des Connaissances (EGC 2007), Revue des Nouvelles Technologies de l'Information (RNTI), Cépaduès-éditions, volume II, pages 539-544, Namur, Belgique, January 23-26, 2007.

16. ROSSI, Fabrice, DE CARVALHO, Francisco, LECHEVALLIER, Yves, **DA SILVA, Alzennyr**. *Dissimilarities for Web Usage Mining.* In : X Conference of the International Federation of Classification Societies (IFCS 2006), Ljubljana, pages 39-46, ISSN : 1431-8814, ISBN : 978-3-540-34415-5, Slovenia, 2006.

17. ROSSI, Fabrice, DE CARVALHO, Francisco, LECHEVALLIER, Yves, **DA SILVA, Alzennyr**. *Comparaison de dissimilarités pour l'analyse de l'usage d'un site Web.* In : G. Ritschard and C. Djeraba, editors, 6es Journées Francophones Extraction et de Gestion des Connaissances (EGC 2006), Revue de Nouvelles Technologies de l'Information (RNTI-E-6), pages 409-414, Lille, France, 2006.

Articles publiés dans des ateliers

18. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves. *Simulation et détection de l'évolution des données temporelles issues de l'usage du Web.* In Actes de l'atelier A.1 : Fouille de données temporelles et analyse de flux de données, 8èmes journées Extraction et Gestion des Connaissances (EGC 2009), p. A.1-17 A.1-23, Strasbourg, 27 January, 2009.

19. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Classifications non supervisées de données évolutives : application au Web Usage Mining.* In Atelier N°4 : Flux de données, 7èmes journées d'Extraction et Gestion des Connaissances (EGC 2007), p. 31-40, Namur, Belgique, January 23, 2007.

20. CHELCEA, Sergiu, **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, TANASA, Doru Tanasa, TROUSSE, Brigitte. *Prétraitement et classification de données complexes dans le domaine du commerce électronique.* In : O. Boussaid and B. Trousse editors, editors, VI atelier : Fouille de Données Complexes dans un processus d'extraction de connaissances, EGC 2006, p. 51-64, Lille, France, 2006.

21. **DA SILVA, Alzennyr**, DE CARVALHO, Francisco, LECHEVALLIER, TROUSSE, Brigitte. *Uma Abordagem para Descoberta do Perfil da Clientela em Web Sites Institucionais.* In Proceedings of the first ECML/PKDD Workshop on Data Mining for Business (DMBiz'05), held in conjunction with the 16th European Conference on Machine Learning (ECML'05) and the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'05), Porto, Portugal, 2005.

22. CHELCEA, Sergiu, **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, TANASA, Doru Tanasa, TROUSSE, Brigitte. *Pre-Processing and Clustering Complex Data in E-Commerce Domain.* In : Proceedings of the First International Workshop on Mining Complex Data 2005 (IEEE MCD'2005), held in conjunction with the Fifth IEEE International Conference on Data Mining (ICDM 2005), Houston, USA, 2005.

23. CHELCEA, Sergiu, **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, TANASA, Doru Tanasa, TROUSSE, Brigitte. *Benefits of InterSite Pre-Processing and Clustering Methods in E-Commerce Domain.* In : Petr Berka and Bruno Crémilleux editors, Proceedings of the ECML/PKDD2005 Discovery Challenge, A Collaborative Effort in Knowledge Discovery from Databases, p. 15-21, Porto, Portugal, 2005.

Autres genres de publications

24. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves. *Analyse exploratoire des indices pour la détermination du bon nombre de clusters : application aux données évolutives.* Groupe de Travail EGC sur la Fouille de Données Complexes (GT EGC-FDC) : Complexité liée aux données multiples. CNAM-Paris, France. 18 june 2009.

25. ANLI, Abdouroihamane, **DA SILVA, Alzennyr**, LECHEVALLIER, Yves. *Spécification et développement du module d'analyse de l'usage.* Technical report, Livrable SP8.2 Projet Eiffel, INRIA, Rocquencourt, 2 September 2008.

26. **DA SILVA, Alzennyr**. *Stratégies de classification non supervisée basées sur fenêtres superposées dans le Web Usage Mining.* In : Ecol'IA 2008, Hammamet, Tunisie, March 20-22, 2008.

27. **DA SILVA, Alzennyr**, LECHEVALLIER, Yves, ROSSI, Fabrice, DE CARVALHO, Francisco. *Clustering of Evolving Data on Web Usage Mining.* In Statistics for Data Mining, Learning and Knowledge Extraction (IASC 07), Aveiro, Portugal, August 30 - September 1, 2007.

28. **DA SILVA, Alzennyr**. *Classification automatique en Web Usage Mining.* In : 2èmes Rencontres Inter-Associations (RIAs'2006) sur la classification et ses applications, Lyon, France, March 20-21, 2006.

Bibliographie

[1] A. Aamodt and E. Plaza, "Case-based reasoning : foundational issues, methodological variations, and system approaches," *AI Communications*, vol. 7, no. 1, pp. 39–59, 1994.

[2] M. Abrams, C. R. Standridge, G. Abdulla, S. Williams, and E. A. Fox, "Caching proxies : Limitations and potentials," Blacksburg, VA, USA, Tech. Rep., 1995.

[3] S. Acharyya and J. Ghosh, "Context-sensitive modeling of web-surfing behaviour using concept trees," in *Proceedings of the Fifth WEBKDD workshop : Webmining as a Premise to Effective and Intelligent Web Applications (WEBKDD 2003)*, Washington, USA, 2003.

[4] C. C. Aggarwal, "On change diagnosis in evolving data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 5, pp. 587–600, 2005.

[5] C. C. Aggarwal and P. S. Yu, "A framework for clustering massive text and categorical data streams," in *Proceedings of the Sixth SIAM International Conference on Data Mining, April 20-22, 2006, Bethesda, MD, USA*, J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava, Eds. SIAM, 2006.

[6] R. Agrawal and R. Srikant, "Mining sequential patterns," *11th International Conference on Data Engineering (ICDE'95)*, pp. 3–14, 1995.

[7] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, P. Buneman and S. Jajodia, Eds. Washington, D.C. : ACM Press, 1993, pp. 207–216.

[8] A. Anli, A. Da Silva, and Y. Lechevallier, "Spécification et développement du module d'analyse de l'usage," INRIA, Rocquencourt, Livrable SP8.2 Projet Eiffel, 2 September 2008.

[9] A. Anli, Z. Jrad, and Y. Lechevallier, "Une approche incrémentielle dans le prétraitement de fichiers logs web : Application à l'analyse des usages d'un site web touristique sémantique," in *Atelier Intégration, interrogation et analyse de LOgs (ILO'08). XXVIème Congrès INFORSID*, Fontainebleau, May 2008.

[10] F. B. Baker and L. J. Hubert, "Measuring the power of hierarchical cluster analysis," *Journal of the American Statistical Association*, vol. 70, no. 349, pp. 31–38, 1975. [Online]. Available : http://www.jstor.org/stable/2285371

[11] P. Baldi, P. Frasconi, and P. Smyth, *Modeling the Internet and the Web : Probabilistic Methods and Algorithms.* Wiley, 2003.

[12] S. Baron and M. Spiliopoulou, "Monitoring change in mining results," in *Data Warehousing and Knowledge Discovery (DaWaK)*, ser. Lecture Notes in Computer Science, Y. Kambayashi, W. Winiwarter, and M. Arikawa, Eds. Springer, 2001, vol. 2114, pp. 51–60.

[13] ——, "Monitoring the evolution of web usage patterns," in *European Web Mining Forum (EWMF)*, ser. Lecture Notes in Computer Science, B. Berendt, A. Hotho, D. Mladenic, M. van Someren, M. Spiliopoulou, and G. Stumme, Eds. Springer, 2003, vol. 3209, pp. 181–200.

[14] S. Baron, M. Spiliopoulou, and O. Günther, "Efficient monitoring of patterns in data mining environments," in *Advances in Databases and Information Systems, 7th East European Conference, ADBIS 2003, Dresden, Germany, September 3-6, 2003, Proceedings*, ser. Lecture Notes in Computer Science, L. A. Kalinichenko, R. Manthey, B. Thalheim, and U. Wloka, Eds., vol. 2798. Springer, 2003, pp. 253–265.

[15] I. Bartolini, P. Ciaccia, I. Ntoutsi, M. Patella, and Y. Theodoridis, "A unified and flexible framework for comparing simple and complex patterns," in *PKDD '04 : Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*. New York, NY, USA : Springer-Verlag New York, Inc., 2004, pp. 496–499.

[16] B. Berendt, B. Mobasher, M. Spiliopoulou, and M. Nakagawa, "The impact of site structure and user environment on session reconstruction in web usage analysis," in *Proceedings of the 4th WebKDD 2002 Workshop, at the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2002)*, Edmonton, Alberta, Canada, July 2002.

[17] P. Bertrand and F. Goupil, "Descriptive statistics for symbolic data," *(Chapter 6), in : Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, (Vol. 15), H.H. Bock and E. Diday, eds, Series : Studies in Classification, Data Analysis, and Knowledge Organisation, Springer-Verlag : Berlin*, pp. 106–119, 2000.

[18] H.-H. Bock and E. Diday, *Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data.* Heidelberge : Springer Verlag, 2000.

[19] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis : Forecasting and Control*, 3rd ed. Prentice Hall, 1994.

[20] P. Brito, "Hierarchical and pyramidal clustering with complete symbolic objects," *(Chapter 11), in : Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, (Vol. 15), H.H. Bock and E. Diday, eds, Series : Studies in Classification, Data Analysis, and Knowledge Organisation, Springer-Verlag : Berlin*, pp. 312–322, 2000.

[21] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, "Model-based clustering and visualization of navigation patterns on a web site," *Data Mining and Knowledge Discovery*, vol. 7, no. 4, pp. 399–424, 2003.

[22] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics - Theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974. [Online]. Available : http://dx.doi.org/10.1080/03610927408827101

[23] L. D. Catledge and J. E. Pitkow, "Characterizing browsing strategies in the world-wide web," *Computer Networks and ISDN Systems*, vol. 27, no. 6, pp. 1065–1073, 1995.

[24] P. Cazes, A. Chouakria, E. Diday, and Y. Schektman, "Extension de l'analyse en composantes principales à des données de type intervalle," *Revue de Statistique appliquée*, vol. 3, pp. 5–24, 1997.

[25] G. Celeux, E. Diday, G. Govaert, Y. Lechevallier, and H. Ralambondrainy, *Classification automatique des données.* Dunod, Paris, 1989.

[26] S. Chakrabarti, S. Sarawagi, and B. Dom, "Mining surprising patterns using temporal description length," in *24th International Conference on Very Large databases (VLDB'98)*, A. Gupta, O. Shmueli, and J. Widom, Eds. Morgan Kaufmann, 1998, pp. 606–617.

[27] M. Chavent, "Criterion-based divisive clustering for symbolic data," *(Chapter 11), in : Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, (Vol. 15), H.H. Bock and E. Diday, eds, Series : Studies in Classification, Data Analysis, and Knowledge Organisation, Springer-Verlag : Berlin*, pp. 299–309, 2000.

[28] M. Chavent, C. Lacomblez, and B. Patouille, "Critère de rand asymétrique," in *Actes des 8èmes Rencontres de la Société Francophone de Classification (SFC01)*, Point-à-Pitre, France, 2001, pp. 82–88.

[29] M. Chavent, "Analyse de données symboliques. une méthode divisive de classification," Thèse de doctorat, Université Paris IX Dauphine, 1997.

[30] X. Chen and I. Petrounias, "Mining temporal features in association rules," *PKDD'99 : Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 295–300, 1999.

[31] A. Chouakria, P. Cazes, and E. Diday, "Symbolic principal component analysis," *in : (Chapter 9) of Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, (Vol. 15), H.H. Bock and E. Diday, eds, Series : Studies in Classification, Data Analysis, and Knowledge Organisation, Springer-Verlag : Berlin*, pp. 200–212, 2000.

[32] R. Cooley, B. Mobasher, and J. Srivastava, "Data preparation for mining world wide web browsing patterns," *Journal of Knowledge and Information Systems*, vol. 1, no. 1, pp. 5–32, 1999.

[33] J. M. Corchado and B. Lees, "Adaptation of cases for case based forecasting with neural network support," in *Soft computing in case based reasoning.* London, UK : Springer-Verlag, 2001, pp. 293–319.

[34] R. Cormack, "A review of classification," *Journal of the Royal Statistical Society. Series A (General)*, vol. 134, no. 3, pp. 321–367, 1971.

[35] F. Corvaisier, A. Mille, and J. M. Pinon, "Information retrieval on the world wide web using a decision making system," in *Proceedings of the Computer-Assisted Searching on the Internet (RIAO 97)*, Montreal, Canada, 1997, pp. 284–295.

[36] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-1, no. 2, pp. 224–227, 1979. [Online]. Available : http://dx.doi.org/10.1109/TPAMI.1979.4766909

[37] A. Debregeas and G. Hébrail, "Interactive interpretation of kohonen maps applied to curves," in *KDD*, 1998, pp. 179–183.

[38] A. P. Dempster, N. M. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, vol. 1, no. 39, pp. 1–38, 1977.

[39] P. Desikan and J. Srivastava, "Mining temporally evolving graphs," *In Proceedings of sixth WebKDD workshop : Web Mining and Web Usage Analys, in conjunction with the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)*, pp. 13–22, 2004.

[40] E. Diday, "Une nouvelle méthode en classification automatique et reconnaissance des formes la méthode des nuées dynamiques." *Revue de Statistique Appliquée*, vol. 19, no. 2, pp. 19–33, 1971.

[41] ——, "Classification automatique séquentielle pour grands tableaux," *Revue Française d'Automatique, Informatique et Recherche Opérationnelle (RAIRO)*, no. B-1, pp. 29–61, 1975.

[42] E. Diday and J. C. Simon, "Clustering analysis," *Communication and Cybernetics 10, Digital Pattern Recognition,K. S. FU, Springer Verlag*, vol. 19, pp. 47–94, 1976.

[43] E. Diday and M. Noirhomme-Fraiture, *Symbolic Data Analysis and the SODAS Software*. New York, NY, USA : Wiley-Interscience, 2008.

[44] O. Elemento, "Apport de l'analyse en composantes principales pour l'initialisation et la validation de cartes topologiques de kohonen," in *Actes des 7èmes journées de la Société Francophone de Classification (SFC'99)*, Nancy, France, 1999.

[45] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *2ndInternational Conference on Knowledge Discovery and Data Mining (KDD-96)*, E. Simoudis, J. Han, and U. Fayyad, Eds. Portland, Oregon : AAAI Press, 1996, pp. 226–231.

[46] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*. AAAI Press - The MIT Press, 1996.

[47] F. Fdez-Riverola and J. M. Corchado, "Using instance-based reasoning systems for changing environments forecasting," in *Workshop on Applying case-based reasoning to time series prediction in conjunction with ICCBR 2003*, Trondheim, Norway, 2003, pp. 219–228.

[48] C. Fraley and A. Raftery, "How many clusters ? which clustering method ? answers via model based cluster analysis," *Computer Journal*, vol. 41, pp. 578–588, 1998.

[49] V. Ganti, J. Gehrke, and R. Ramakrishnan, "Demon : Mining and monitoring evolving data," in *IEEE Transactions on Knowledge and Data Engineering*, 2000, pp. 439–448.

[50] V. Ganti, J. Gehrke, R. Ramakrishnan, and W.-y. Loh, "A framework for measuring changes in data characteristics," in *In PODS*. ACM Press, 1999, pp. 126–137.

[51] L. A. Goodman and W. H. Kruskal, "Measures of association for cross classification," *Journal of the American Statistical Association*, vol. 49, pp. 732–64, 1954.

[52] J. A. Hartigan, *Clustering Algorithms.* New York, NY, USA : John Wiley & Sons, Inc., 1975. [Online]. Available : http://main.cs.qub.ac.uk/~fmurtagh/class/csna/re_CD/CD2002/data/Hartigan/contents.pdf

[53] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[54] L. J. Hubert and J. R. Levin, "The name assigned to the document by the author. this field may also contain sub-titles, series names, and report numbers. a general statistical framework for assessing categorical clustering in free recall," *Psychological Bulletin*, no. 83, pp. 1072–1080, 1976. [Online]. Available : http://www.eric.ed.gov/ERICWebPortal/contentdelivery/servlet/ERICServlet?accno=ED116162

[55] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* San Francisco, CA : ACM Press, 2001, pp. 97–106.

[56] M. Jaczynski and B. Trousse, "Broadway : A case-based system for cooperative information browsing on the world-wide-web," in *Collaboration between Human and Artificial Societies, Coordination and Agent-Based Distributed Computing.* London, UK : Springer-Verlag, 1999, pp. 264–283.

[57] T. Kohonen, *Self-Organizing Maps*, 3rd ed., ser. Springer Series in Information Sciences. Springer, 1995, vol. 30, last edition published in 2001.

[58] J. Kolodner, *Case-Based Reasoning.* 2929 Campus Drive, suite260, SanMateo, CA, USA : Morgan Kaufmann Publishers, Inc., 1993.

[59] R. Kosala and H. Blockeel, "Web mining research : A survey," *ACM SIGKDD Explorations : Newsletter of the Special Interest Group on Knowledge Discovery and Data Mining*, vol. 2, pp. 1–15, 2000.

[60] W. J. Krzanowski and Y. T. Lai, "A criterion for determining the number of groups in a data set using sum-of-squares clustering," *Biometrics*, vol. 44, no. 1, pp. 23–34, March 1988.

[61] S. K. Lam and J. Riedl, "Shilling recommender systems for fun and profit," in *WWW*, S. I. Feldman, M. Uretsky, M. Najork, and C. E. Wills, Eds. ACM, 2004, pp. 393–402.

[62] S. Laxman and P. S. Sastry, "A survey of temporal data mining," *SADHANA - Academy Proceedings in Engineering Sciences, Indian Academy of Sciences*, vol. 31, no. 2, pp. 173–198, 2006.

[63] L. Lebart, *Méthodes Factorielles.* Dunod, 1997.

[64] L. Lebart, A. Morineau, and M. Piron, *Statistique exploratoire multidimensionnelle.* Dunod, 1995.

[65] Y. Lechevallier, "Optimisation de quelques critères en classification automatique et application à l'étude des modifications des protéines en pathologie clinique," Thèse de doctorat, Université Paris VI, 1974.

[66] H. Lieberman, "Letizia : An agent that assists web browsing," in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, C. S. Mellish, Ed. Montreal, Quebec, Canada : Morgan Kaufmann publishers Inc. : San Mateo, CA, USA, 1995, pp. 924–929.

[67] B. Liu, Y. Ma, and R. Lee, "Analyzing the interestingness of association rules from the temporal dimension," *ICDM '01 : Proceedings of the 2001 IEEE International Conference on Data Mining*, pp. 377–384, 2001.

[68] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. University of California Press, 1967, pp. 281–297.

[69] M. Malek and R. Kanawati, "Cobra : A cbr-based approach for predicting users actions in a web site," *Case-Based Reasoning Research and Development : 4th International Conference on Case-Based Reasoning (ICCBR 2001)*, pp. 336–346, 2001.

[70] A. Marascu and F. Masseglia, "Extraction de motifs séquentiels dans les flots de données d'usage du web." in *Actes des sixièmes journées Extraction et Gestion des Connaissances (EGC 2006)*, ser. Revue des Nouvelles Technologies de l'Information (RNTI-E-6), G. Ritschard and C. Djeraba, Eds. Lille, France : Cépaduès-Éditions, 2006, pp. 627–638.

[71] F. Masseglia, D. Tanasa, and B. Trousse, "Diviser pour découvrir. une méthode d'analyse du comportement de tous les utilisateurs d'un site web," *Ingénierie des Systèmes d'Information*, vol. 9, no. 1, pp. 61–83, 2004.

[72] Q. Mei and C. Zhai, "Discovering evolutionary theme patterns from text : an exploration of temporal text mining," in *KDD '05 : Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining.* New York, NY, USA : ACM, 2005, pp. 198–207.

[73] A. Micarelli and F. Sciarrone, "A case-based system for adaptive hypermedia navigation," *Advances in Case-Based Reasoning, Proc. of the 3rd European Workshop on Case-Based Reasoning (EWCBR'96)*, vol. 1168, pp. 266–279, 1996.

[74] G. W. Milligan and M. C. Cooper, "Study of the comparability of external criteria for hierarchical cluster analysis," *Multivariate Behavioral Research*, vol. 21, no. 4, pp. 441 – 458, 1986.

[75] B. Mobasher, "Web usage mining and personalization," in *The Practical Handbook of Internet Computing.* Florida, USA : Chapman Hall & CRC Press, 2004.

[76] R. Moskovitch and Y. Shahar, "Temporal data mining based on temporal abstractions," *Proceedings of the 2nd Workshop on Temporal Data Mining (TDM 2005), held in conjunction with the 5th IEEE International Conference on Data Mining (ICDM'05)*, 27 November 2005.

[77] D. B. Neill, A. W. Moore, M. Sabhnani, and K. Daniel, "Detection of emerging space-time clusters," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, R. Grossman, R. J. Bayardo, and K. P. Bennett, Eds. ACM, 2005, pp. 218–227.

[78] M. Noirhomme-Fraiture and M. Rouard, "Visualisation de données multivariées : évaluation de la représentation en étoile zoom," *IHM 98*, pp. 121–126, 1998.

[79] ——, "Visualizing and editing symbolic objects," *(Chapter 7), in : Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, (Vol. 15), H.H. Bock and E. Diday, eds, Series : Studies in Classification, Data Analysis, and Knowledge Organisation, Springer-Verlag : Berlin*, pp. 125–138, 2000.

[80] M. Noirhomme-Fraiture, "Multimedia support for complex multidimensional data mining," *Workshop on Multimedia Data Mining, KDD'2000*, pp. 54–59, 2000.

[81] D. Pierrakos, G. Paliouras, C. Papatheodorou, and C. D. Spyropoulos, "Web usage mining as a tool for personalization : A survey," *User Modeling and User-Adapted Interaction*, vol. 13, no. 4, pp. 311–372, 2003.

[82] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[83] J. F. Roddick and M. Spiliopoulou, "A survey of temporal knowledge discovery paradigms and methods," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 4, pp. 750–767, 2002.

[84] P. Rousseeuw, "Silhouettes : a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.

[85] M. Samia, "Temporal web mining." in *Grundlagen von Datenbanken*, H. Höpfner, G. Saake, and E. Schallehn, Eds. Fakultät für Informatik, Universität Magdeburg, 2003, pp. 27–31.

[86] M. Samia and S. Conrad, "From temporal data mining and web mining to temporal web mining," *Sixth International Baltic Conference on Databases and Information Systems (BalticDB&IS'2004)*, pp. 232–245, 2004.

[87] G. Saporta, *Probabilités, analyse des données et statistique.* Technip, 1990.

[88] D. M. Sow, D. P. Olshefski, M. Beigi, and G. Banavar, "Prefetching based on web usage mining." in *Middleware*, ser. Lecture Notes in Computer Science, M. Endler and D. C. Schmidt, Eds. Springer, 2003, vol. 2672, pp. 262–281.

[89] M. Spiliopoulou, "Data mining for the web," *Workshop on Machine Learning in User Modelling of the ACAI99*, pp. 588–589, 1999.

[90] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult, "Monic : modeling and monitoring cluster transitions," in *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, Eds. ACM, 2006, pp. 706–711.

[91] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan, "Web usage mining : Discovery and applications of usage patterns from web data," *SIGKDD Explorations*, vol. 1, no. 2, pp. 12–23, 2000.

[92] B. S. Suryavanshi, N. Shiri, and S. P. Mudur, "Incremental relational fuzzy subtractive clustering for dynamic web usage profiling," *In Proceedings of WebKDD Workshop on Taming Evolving, Expanding and Multi-faceted Web Clickstreams held in conjunction with the 11th*

ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2005), 2005.

[93] D. Tanasa, "Web usage mining : Contributions to intersites logs preprocessing and sequential pattern extraction with low support," Thèse de doctorat, University of Nice Sophia Antipolis, 3 June 2005.

[94] D. Tanasa and B. Trousse, "Advanced data preprocessing for intersites web usage mining," *IEEE Intelligent Systems*, vol. 19, no. 2, pp. 59–65, March-April 2004.

[95] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001. [Online]. Available : http://dx.doi.org/10.1111/1467-9868.00293

[96] M. Touati, M. C. Rahal, F. Afonso, and E. Diday, "Le logiciel sodas : avancées récentes. un outil permettant d'analyser et de visualiser des données symboliques," in *EGC*, ser. Revue des Nouvelles Technologies de l'Information, F. Guillet and B. Trousse, Eds., vol. RNTI-E-11. Cépaduès-Éditions, 2008, pp. 239–240.

[97] C. J. van Rijsbergen, *Information Retrieval*, 2nd ed. London : Butterworths, 1979.

[98] W3C, "Logging control in w3c httpd," *http ://www.w3.org/TR/WD-logfile*, July 1995.

[99] T. W. Yan, M. Jacobsen, H. Garcia-Molina, and U. Dayal, "From user access patterns to dynamic hypertext linking," in *Proceedings of the fifth international World Wide Web conference on Computer networks and ISDN systems.* Amsterdam, The Netherlands, The Netherlands : Elsevier Science Publishers B. V., 1996, pp. 1007–1014.

[100] H. Yang, S. Parthasarathy, and S. Mehta, "A generalized framework for mining spatio-temporal patterns in scientific data," in *KDD '05 : Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining.* New York, NY, USA : ACM, 2005, pp. 716–721.

[101] G. Youness and G. Saporta, "Une méthodologie pour la comparaison de partitions," *Revue de Statistique Appliquée*, vol. 52, no. 1, pp. 97–120, 2004.

[102] F. Zehraoui and Y. Bennani, "M-som : matricial self organizing map for sequence clustering and classification," *IEEE International Joint Conference on Neural Networks (IJCNN 2004)*, vol. 1, pp. 763–768, 2004.

[103] ——, "Som-art : Incorporation des propriétés de plasticité et de stabilité dans une carte auto-organisatrice," *Atelier : Fouille de données complexes dans un processus d'extraction des connaissances (FDC) des 4ème journées Extraction et Gestion des Connaissances (EGC 2004)*, pp. 169–180, 2004.

[104] F. Zehraoui, R. Kanawati, and S. Salotti, "Case base maintenance for improving prediction quality," *Case-Based Reasoning Research and Development*, vol. 2689, pp. 703–717, 2003.

[105] ——, "Cbr system for sequence prediction : Casep," in *Proceedings of the International workshop on applying case-based reasoning on time series prediction*, Trondhiem, Norway, 2003, pp. 260–269.

[106] ——, "Casep2 : Hybrid case-based reasoning system for sequence processing," *Advances in Case-Based Reasoning*, vol. 3155, pp. 449–463, 2004.

[107] B. Zhou, S. C. Hui, and A. C. M. Fong, "Discovering and visualizing temporal-based web access behavior," in *Proceedings of the the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. Washington, DC, USA : IEEE Computer Society, 2005, pp. 297–300.

Résumé

Le nombre d'accès aux pages Web ne cesse de croître. Le Web est devenu l'une des plates-formes les plus répandues pour la diffusion et la recherche d'information. Par conséquence, beaucoup d'opérateurs de sites Web sont incités à analyser l'usage de leurs sites afin d'améliorer leur réponse vis-à-vis des attentes des internautes. Or, la manière dont un site Web est visité peut changer en fonction de divers facteurs. Les modèles d'usage doivent ainsi être mis à jour continuellement afin de refléter fidèlement le comportement des visiteurs. Ceci reste difficile quand la dimension temporelle est négligée ou simplement introduite comme un attribut numérique additionnel dans la description des données. C'est précisément sur cet aspect que se focalise la présente thèse. Pour pallier le problème d'acquisition des données réelles d'usage, nous proposons une méthodologie pour la génération automatique des données artificielles permettant la simulation des changements. Guidés par les pistes nées des analyses exploratoires, nous proposons une nouvelle approche basée sur des fenêtres non recouvrantes pour la détection et le suivi des changements sur des données évolutives. Cette approche caractérise le type de changement subi par les groupes de comportement (apparition, disparition, fusion, scission) et applique deux indices de validation basés sur l'extension de la classification pour mesurer le niveau des changements repérés à chaque pas de temps. Notre approche est totalement indépendante de la méthode de classification et peut être appliquée sur différents types de données autres que les données d'usage. Des expérimentations sur des données artificielles ainsi que sur des données réelles issues de différents domaines (académique, tourisme et *marketing*) ont été réalisées pour l'évaluer l'efficacité de l'approche proposée.

Mots-clés : Analyse de données, classification non supervisée, données évolutives, fouille d'usage du Web.

Abstract

Nowadays, more and more organizations are becoming reliant on the Internet. The Web has become one of the most widespread platforms for information change and retrieval. The growing number of traces left behind user transactions (e.g. : customer purchases, user sessions, etc.) automatically increases the importance of usage data analysis. Indeed, the way in which a web site is visited can change over time. These changes can be related to some temporal factors (day of the week, seasonality, periods of special offer, etc.). By consequence, the usage models must be continuously updated in order to reflect the current behaviour of the visitors. Such a task remains difficult when the temporal dimension is ignored or simply introduced into the data description as a numeric attribute. It is precisely on this challenge that the present thesis is focused. In order to deal with the problem of acquisition of real usage data, we propose a methodology for the automatic generation of artificial usage data over which one can control the occurrence of changes and thus, analyse the efficiency of a change detection system. Guided by tracks born of some exploratory analyzes, we propose a tilted window approach for detecting and following-up changes on evolving usage data. In order measure the level of changes, this approach applies two external evaluation indices based on the clustering extension. The proposed approach also characterizes the changes undergone by the usage groups (e.g. appearance, disappearance, fusion and split) at each timestamp. Moreover, the refereed approach is totally independent of the clustering method used and is able to manage different kinds of data other than usage data. The effectiveness of this approach is evaluated on artificial data sets of different degrees of complexity and also on real data sets from different domains (academic, tourism and marketing).

Keywords : Clustering, data analysis, evolving data, Web Usage Mining (WUM).