# Système de recommandation équitable d'oeuvres numériques. En quête de diversité

Pierre-René Lherisson

N°d'ordre NNT : 2018LYSES018

**THESE de DOCTORAT DE L'UNIVERSITE DE LYON**
opérée au sein du
**Laboratoire Hubert Curien**

**École Doctorale** ED SIS 488
**École Doctorale Sciences, Ingénierie et Santé**

**Discipline : Informatique**:

Soutenue publiquement le 20/06/2018, par :
**Pierre-René Lhérisson**

---

# Système de recommandation équitable d'œuvres numériques
## En quête de diversité

---

Devant le jury composé de :

Rousseaux, Francis Prof. des Universités Université de Reims Rapporteur
Velcin, Julien Maître de Conférence HDR Université Lumière Lyon 2 Rapporteur
Negre, Elsa Maître de Conférence HDR Université Paris-Dauphine Examinatrice
Pottier, Laurent Prof. des Universités Université Jean Monnet Saint-Étienne Examinateur

Maret, Pierre Prof. des Universités Université Jean Monnet Saint-Étienne Directeur de thèse
Muhlenbach, Fabrice Maître de Conférence Université Jean Monnet Saint-Étienne Co-directeur de thèse
Claquin, Cédric Directeur-adjoint 1D Lab Invité

# RÉSUMÉ

Les systèmes de recommandation jouent un rôle important dans l'orientation des choix des utilisateurs. La recommandation se fait généralement par une optimisation d'une mesure de précision de l'adéquation entre un utilisateur et un produit. Cependant, plusieurs travaux de recherche ont montré que l'optimisation de la précision ne produisait pas les recommandations les plus utiles pour les utilisateurs. Un système trop précis peut contribuer à confiner les utilisateurs dans leur propre bulle de choix. Ceci peut aussi produire un effet de foule qui va concentrer les usages autour de quelques articles populaires. Par conséquent, il y a un manque de diversité et de nouveauté dans les recommandations et une couverture limitée du catalogue. Par ailleurs, l'utilisateur peut ressentir de la frustration envers ces recommandations monotones et arrêter de se fier au système. Ce type de recommandation va à l'antithèse de l'esprit humain qui peut être friand de nouveauté et de diversité. Même si la routine peut être sécurisante, l'être humain aime sortir des sentiers battus pour, par exemple, découvrir de nouveaux produits, tenter de nouvelles expériences. Cette absence de découverte est préjudiciable pour une plateforme numérique, surtout si cette dernière veut être équitable dans ses recommandations envers tous les producteurs de contenu (par exemple, les artistes, les écrivains, les développeurs de jeux vidéos, les vidéastes).

Dans cette thèse, nous présentons deux familles de modèles qui cherchent à produire des résultats qui vont au-delà des aspects de précision pour des systèmes de recommandation pour des produits culturels basés sur le contenu. Les deux modèles que nous présentons reposent sur l'étude du profil de l'utilisateur avant de lui proposer des listes de recommandations contenant des articles nouveaux et divers. Ces approches captent la diversité qu'il y a dans le profil de l'utilisateur et répondent à cette diversité en cherchant à créer une liste diversifiée de recommandations sans trop pénaliser la précision. Le premier modèle repose principalement sur une approche de clustering. Dans ce modèle, nous proposons de la diversité à l'utilisateur tout en restant dans le périmètre de ses goûts. Le second modèle est basé sur une fonction issue de la loi normale. Nous faisons l'hypothèse de l'existence d'une zone intermédiaire définie entre des éléments considérés comme trop similaires et d'autres considérés comme trop différents. Cette zone intermédiaire est une zone propice à la découverte et à l'exploration de genres et d'expériences nouveaux.

Nos propositions sont testées sur des jeux de données standards et comparées à des algorithmes de l'état de l'art. Les résultats de nos expériences montrent que nos approches apportent de la diversité et de la nouveauté et sont compétitives par rapport aux méthodes de l'état de l'art. Nous proposons également une expérience utilisateur pour valider notre modèle basé sur la fonction issue de la loi normale. Les résultats des expériences centrées sur l'utilisateur montrent que ce modèle correspond au comportement cognitif de l'être humain ainsi qu'à sa perception de la diversité.

# ABSTRACT

Recommender systems play a leading role in user's choice guidance. The search of accuracy in such systems is generally done through an optimization of a function between the items and the users. It has been proved that maximizing only the accuracy does not produce the most useful recommendation for the users. This can confine individuals inside the bubble of their own choices. Additionally, it tends to emphasize the agglomaration of the users' behavior on few popular items. Thus, it produces a lack of diversity and novelty in recommendations and a limited coverage of the platform catalog. This can lead to an absence of discovery. Monotony and frustration are also induced for the users. This non-discovery is even more crucial if the platform wants to be fair in its recommendations with all contents' producers (e.g, music artists, writers, video game developers or videographers). The non diversity, and novelty problem is more important for the users because it has been shown that human mind appreciates when moved outside of its comfort zone. For example, the discovery of new artists, the discovery of music genres for which he is not accustomed.

In this thesis we present two families of model that aim to go beyond accuracy in content-based recommender system scenario. Our two models are based on a user profile understanding prior to bring diversification. They capture the diversity in the user profile and respond to this diversity by looking to create a diverse list of recommendation without loosing to much accuracy. The first model is mainly built upon a clustering approach, while the second model is based on an wavelet function. This wavelet function in our model helps us delimit an area where the user will find item slightly different from what he liked in the past. This model is based on the assumption of the existence of a defined intermediate area between similar and different items. This area is also suitable for discovery.

Our proposals are tested on a common experimental design that consider well-known datasets and state-of-the-art algorithm. The results of our experiments show that our approaches indeed bring diversity and novelty and are also competitive against state-of-the-art method. We also propose a user-experiment to validate our model based on the wavelet. The results of user-centered experiments conclude that this model corresponds with human cognitive and perceptual behavior.

# INTRODUCTION

## Motivation : au service de la création indépendante

L'arrivée de l'ère numérique avait laissé présager un avenir radieux pour les articles de niches présents dans les catalogues de commerce en ligne ainsi que pour les artistes à faible audience des sites de streaming. Avec les technologies de la révolution numérique, on pouvait croire que l'accès à un choix qui augmentait allait automatiquement débloquer une demande pour ce choix. Il est vrai que certaines entreprises comme *Netflix* n'ont pas d'entrepôt physique pour stocker leurs produits numériques. Elles ne sont plus limitées par les murs, un disquaire par exemple doit choisir ce qu'il doit mettre en rayon afin d'attirer une clientèle. Les magasins traditionnels ont la plupart du temps fait face à une contrainte économique au niveau de l'emplacement. Chaque espace, chaque mètre-carré, doit être optimisé, car l'emplacement coûte cher. En revanche, quand ces magasins s'affranchissent des contraintes de l'espace physique ils peuvent potentiellement exposer tout le catalogue mondial. Cet étalage du catalogue mondial (comme sur *Amazon*) devrait permettre peu à peu un gain d'intérêt pour les cultures alternatives non majoritaires. Le public se lassera de cette culture de masse pour se décrouvir des passions pour les produits anciens, étrangers, indépendants. De plus, l'avènement d'Internet a libéré les consommateurs que nous sommes du dictat de la radio et de la télévision.

Chris Anderson [Anderson, 2008] prenait l'exemple de son fils, un adolescent nommé Ben, qui en 2008 (date de publication du livre) grâce à Internet et son *iPod* avait accès à tous types de contenus. Il pouvait regarder les films les plus populaires Hollywood tout en jouant aux jeux vidéos créés par des membres de la communauté des joueurs. Il avait aussi accès aux *animes* de la télévision japonaise grâce au réseau de téléchargement de type *peer-to-peer BitTorrent*. Cet accès à la culture des mangas a poussé Ben à étudier le japonais à l'école comme certains de ses amis. Ben écoutait très peu la radio et regardait peu la télévision. Grâce à Internet, il a eu accès à des cultures totalement différentes de ce qui était retransmis à la télévision ou la radio en 2008. Dans un scénario pareil, tout nous pousse à croire que Ben et ses amis pouvaient échanger sur un grand nombre de sujets, tant ils devaient consommer chacun quelque chose de différent. Parallèlement, pour reprendre Anderson, dans les années 50 et 60, il y avait de fortes chances d'affirmer que tout le monde ou presque avait regardé la même chose à la télévision, du bureau à la cour de récréation, la nuit précédente. L'ère de la culture unique, du marché unique pour les vendeurs de culture était terminée, il fallait faire place maintenant à une multitude de marchés, de cultures et sous-cultures, une multitude de choix.

Dans les bases de données, les *bestsellers*, représentant de la culture unique, et les *neversellers*, représentant peut-être les cultures alternatives, ne sont que deux entrées qui sont égales aux yeux

de la technologie et qui ont le même coût au niveau de l'espace de rangement. Donc on peut générer des revenus aussi à partir des articles de niches. Chris Anderson a produit une enquête auprès des leaders de la nouvelle [1] industrie numérique (*Amazon*, *Apple* avec son produit *iTunes*, *Netflix*) et il a pu se rendre compte que :

(i) le marché des *hits* (c.-à-d. des articles les plus populaires) était solide ;

(ii) le marché des produits de niche était porteur.

*Amazon*, *Apple*, *Netflix* affirmaient tous avoir vendu ou loué au moins une fois plus de 95% de leur catalogue. Chacune de ces compagnies était impressionnée par la demande qu'elles observaient pour les catégories de produits jamais mis en avant par l'économie des *hits*, c'est-à-dire la tendance à toujours mettre en avant les produits les plus populaires (*Top 50*, *Top 40*). Il y a donc un vrai potentiel économique en vendant un peu de tout. Chris Anderson avait même prédit une prise de pouvoir des articles de la longue traîne (chapitre 2), et une mort de l'économie et de la culture des *hits*.

Nous rejoignons Anderson pour dire que l'ère digitale a profondément changé la manière d'accéder aux biens culturels. Les œuvres peuvent être partagées à tous les instants, quels qu'en soient les auteurs ou l'origine. L'accroissement et la vitesse de diffusion des biens culturels interrogent ainsi plusieurs aspects sur lesquels nos cultures reposaient jusqu'à présent : la place de l'auteur, les droits associés aux œuvres - et plus largement le concept de propriété- la notion de diversité, l'accès de tous à la culture, etc.

Mais contrairement à ce qui avant était annoncé par Anderson, notamment dans le milieu musical (qui a globalement dû faire face le premier à ces mutations) et plus globalement dans l'ensemble des secteurs créatifs, la diversité disparaît progressivement au profit d'une concentration mécanique - des acteurs comme de la consommation - et une uniformisation progressive des goûts. Dans ce nouvel espace, les grands noms se renforcent économiquement (1% des artistes captaient par exemple en 2013 plus de 77% des revenus du streaming [2]), traduisant un déséquilibre croissant dans le partage de la valeur à l'ère du numérique qui menace à terme le renouvèlement de la création. Cette mort de la longue traîne (chapitre 2) est donc finalement amplifiée par le passage au numérique. Certaines compagnies continuent à utiliser les *blockbusters* pour attirer un public et dégager un profit pour continuer à exister. *Netflix* par exemple a décidé d'arrêter la production d'une série jugée trop couteuse (*Sense8* des Wachowski) par rapport aux profits générés. Cette série produite par *Netflix* bien qu'aimée par les *fans* n'avait pas générée assez de vues aux yeux de la direction du service de *streaming* préférant l'arrêter et axer sa production sur des séries plus populaires.

Face à ces logiques de concentration, un certain nombre d'acteurs (artistes, labels, managers, passionnés de musique...) se sont rassemblés en 2014 pour créer la Scic (société coopérative d'intérêt collectif) 1D Lab [3] : une coopérative souhaitant explorer de nouvelles manières de

1. L'industrie numérique était « nouvelle » en 2008
2. https ://musicindustryblog.wordpress.com/2014/03/04/the-death-of-the-long-tail/
3. `http://1d-lab.eu/`

découvrir et de partager les biens culturels, proposer de nouvelles expériences aux usagers et penser des écosystèmes à la fois durables et animés à minima de notions d'intérêt général [Claquin et Lhérisson, 2016].

L'exploration et l'analyse des processus mis en place par 1D Lab a débouché notamment à la construction de 1D touch [4] (première plateforme mondiale de streaming multicontenus centrée sur la création indépendante). Loin d'être une réponse toute faite aux problèmes associés à l'arrivée de cette fameuse révolution numérique, elle explore de façon collective un certain nombre de démarches et tente d'illustrer par l'exemple des mouvements de fonds qui traversent, de façon parfois invisible, la société du XXI$^{ème}$ siècle : partage équitable de la valeur, nouveaux modes d'échanges entre pairs, notion de communs (*commons*), modèles collaboratifs, etc. *1D touch* est une plateforme de streaming lancée en 2014, accessible par le web, permettant de consulter et de découvrir des ressources numériques culturelles issues de créateurs indépendants. D'abord centrée sur la musique, *1D touch* ambitionne de devenir progressivement la première plateforme internationale de découverte de contenus indépendants issus des secteurs de l'image animée, livre, jeu vidéo, photo. Cet outil est développé selon les priorités d'1D Lab :

(i) renforcer la visibilité et la rémunération des catalogues indépendants sous-exposés dans les réseaux traditionnels des industries créatives ;

(ii) décloisonner les esthétiques artistiques et proposer à une communauté d'usagers des expériences de découverte culturelle multicontenus (musique, jeu vidéo, image animée, livre numérique) ;

(iii) développer des outils réplicables pour accompagner les transitions numériques et favoriser l'apparition de « nouveaux écosystèmes territoriaux ».

Outre 1D touch, 1D Lab a développé une autre application qui incarne aussi les valeurs citées plus haut : *Divercities* [5]. Il s'agit d'une application mobile gratuite (smartphone et tablette) qui permet de repérer, collecter et partager dans l'espace public des « capsules créatives » : sélections multimédias (musique, vidéo, livre et BD numériques, jeu vidéo) composées de contenus culturels issus de la sphère indépendante, d'artistes reconnus comme de créateurs émergents. L'idée est de promouvoir au travers d'une expérience ludique et mobile la vitalité créative et culturelle locale et la qualité de son maillage de lieux et de compétences (à la fois professionnels et amateurs). Ce dispositif est également l'occasion d'interroger et de développer de façon innovante les usages touristiques sur un territoire et ainsi d'en renforcer l'attractivité et l'image.

1D Lab intègre aussi dans le processus d'amélioration de ses produits une dimension *laboratoire* d'innovation. Par ce côté laboratoire, 1D Lab s'inscrit dans de nombreux réseaux pour une captation dynamique des enjeux sociétaux des territoires et des producteurs culturels ainsi que sur les technologies offrant des opportunités de solutions [6]. Dans ces optiques 1D Lab, a

4. http://1dtouch.com/
5. http://divercities.eu/
6. http://1d-lab.eu/laboratoire-dexperimentations

collaboré avec l'équipe Connected Intelligence [7] du laboratoire Hubert-Curien [8] de Saint-Etienne. De cette collaboration est née un projet de thèse CIFRE qui a débouché sur des recherches présentées dans le document que vous êtes en train de lire.

# Problématique : l'allégorie de la caverne numérique

L'ère numérique a changé les nouveaux rapports à la culture, et a permis à la culture *mainstream* de s'affirmer. L'ère numérique a aussi produit la surinformation ou l'infobésité (en anglais « *information overload* »). Ce concept a été introduit pour décrire le sentiment de fatigue et de confusion ressenti par une personne quand elle doit traiter un surplus d'information. Le terme a été popularisé par l'écrivain, sociologue et futurologue américain Alvin Toffler [Toffler, 1971]. Actuellement, l'une des premières sources d'infobésité est le *web 2.0* et le phénomène des *Big Data* [Ho et Tang, 2001]. Comme il y a eu l'âge de pierre, et l'âge du pétrole, maintenant nous vivons l'âge de l'information. Un peu plus de 50% de la population mondiale est connectée maintenant à Internet [9]. On parlait de 40% en 2014 (2014 est l'année où cette thèse a débuté). Cette augmentation de la population connectée fait aussi augmenter exponentiellement les données générées. Plusieurs observateurs du web se sont mis d'accord pour dire que la taille de l'univers digital va doubler tous les deux ans. Pour illustrer nos propos, nous avons observé en 1 seconde sur Internet, 7 789 *tweets* générés, 63 026 recherches effectuées sur *Google*, 71 133 vidéos regardées sur *Youtube*, 2 631 668 emails envoyés, 807 photos postées sur *Instagram*, 2 804 appels passés sur *Skype*, et surtout 49 384 Go de trafic Internet générés. Chacun peut observer des chiffres similaires sur ce site *Internet Live Stats* [10]. Les sites de e-commerce ne sont pas non plus en reste : *Amazon* propose 300 millions de produits à ses clients, *Spotify* et *Apple Music* proposent tous les deux 30 millions de titres, *Deezer* propose 35 millions de titres, *Netflix* propose 6000 titres aux États-Unis (la taille du catalogue varie en fonction du pays). Le challenge est donc de trouver des moyens pour accéder rapidement à l'information nécessaire dans cette masse où l'utilisateur se trouve souvent désarmé. Parmi les solutions proposées, les technologies de recommandations, en cherchant à fournir des suggestions personnalisées dans ces vastes catalogues, ont prouvé qu'elles pouvaient apporter aux utilisateurs ce qu'ils recherchent et de plus elles sont des sources majeures de revenus (35% des revenus d'*Amazon* proviennent de leur système de recommandation). Ces technologies permettent de filtrer l'information qui arrive en trop grande quantité pour l'utilisateur et par la même le guider dans ses choix. Parallèlement, nous constatons une sur-concentration de l'attention autour de certaines informations qui gagnent une immense, soudaine et brève popularité en raison des effets de coordination virale qui orientent les publics vers quelques produits [Mellet, 2009]. Nous pouvons avancer que certains de ces systèmes souffrent d'un biais sur la popularité. En effet, pour faire une prédiction pour un

7. https://connected-intelligence.univ-st-etienne.fr/
8. https://laboratoirehubertcurien.univ-st-etienne.fr/en/index.html
9. http://www.internetworldstats.com/stats.htm
10. http://www.internetlivestats.com/one-second/

utilisateur, ces systèmes se basent souvent sur le passé de ceux qui ont eu un comportement similaire à cet utilisateur. Or dans ces bases de données d'activité d'utilisateurs, certains produits n'ont pas généré assez de vues pour apparaitre dans les recommandations. Ainsi à force d'être cités par tous, les plus reconnus deviennent aussi les plus populaires et recoivent en conséquence le plus de clics [Hindman, 2008]. De plus, certaines entreprises ne cherchent pas non plus à mettre en avant des produits peu connus (comme les artistes indépendants). Le système de recommandation d'une entreprise comme *Amazon* n'est sans doute pas optimisé pour épouser la noble cause de faire connaitre les artistes (musicaux) indépendants. Suivant une logique marchande capitaliste, l'entreprise est animée par des objectifs financiers l'amenant à vendre le maximum quitte à vendre le même produit à tout le monde. Pour une entreprise qui se veut équitable, il est important de créer des algorithmes qui ne produisent pas ce genre d'effet.

Les algorithmes de recommandation font aussi confiance aux régularités des structures de goûts des utilisateurs pour produire les recommandations. Car il existe un caractère régulier et prévisible dans les comportements des utilisateurs [Cardon, 2015]. Pour le sociologue Pierre Bourdieu, *l'habitus* est cette disposition incorporée à travers laquelle la société façonne des choix réguliers et prévisibles jusque dans les petites infractusuosités du quotidien [Bourdieu, 1980]. Selon cette théorie, les gens auraient les mêmes comportements sur les divers sites de streaming, et l'algorithme en raison de son architecture ne pourra donc pas produire de recommandations qui sortent des consommations des utilisateurs. Peut-on rendre les algorithmes responsables de leurs résultats quand ils ne font que recopier le comportement des utilisateurs ? Par ailleurs, il existe un décalage de plus en plus important entre ce que les utilisateurs disent faire et ce qu'ils font réellement : c'est la multiplication des désirs d'être et la réalité des existences quotidiennes [Rosa et Renault, 2013]. Nous pouvons répondre à la question précédente en avançant que les algorithmes poussent les utilisateurs à répéter leurs mêmes comportements à la façon du film *Un jour sans fin* d'Harold Ramis [11] en faisant toujours l'hypothèse que le futur sera une reproduction du passé. Comme l'indique Cardon [2015]

> Cette manière d'utiliser l'information enferme les individus dans la bulle de leur choix, plie leur destin dans l'entonnoir du probable et alimente la précision de leurs algorithmes de ciblage au prix d'une capture disproportionnée d'informations personnelles

.

De façon métaphorique, nous pouvons dire que les utilisateurs se retrouvent au fond d'une caverne numérique. Ils y sont placés en raison de la nature de leurs opinions, et peuvent aussi être maintenus dans cette caverne par les algorithmes de recommandation. À l'extérieur, on retrouve le monde de la connaissance représenté par l'ensemble des autres produits provenant de genres, de cultures différentes. On retrouve aussi les commentaires différents d'autres internautes. Il faut donc pour l'algorithme présenter à l'utilisateur d'autres visions du monde, de manière à ce que ce dernier opère une révolution sur lui, et convertisse son regard sur le monde. Notre

11. Groundhog Day sorti en 1993 avec Bill Murray

allégorie de la caverne numérique est une invitation à créer des algorithmes pour pousser les utilisateurs, sans les perdre et les désorienter, à sortir de la prison numérique dans laquelle ils se trouvent pour aller vers des expériences nouvelles et inédites. De nouveaux diamants, culturels, informationnels ou sociaux, peuvent être trouvés n'importe où, mais pour cela, il faut aller les chercher au-delà du périmètre de notre univers connu et de notre zone de confort.

## Objectifs de recherche

Le but de cette thèse est de proposer des approches qui vont apporter de la nouveauté et de la diversité dans les listes de recommandation. Nous pensons que la nouveauté et la diversité sont des dimensions qui nous permettront de nous assurer que les recommandations proposées ne sont pas monotones et qu'elles n'enferment pas l'utilisateur dans un périmètre donné. Nous proposons des méthodes qui vont pousser l'utilisateur à aller au-delà de son territoire balisé, sa bulle de filtre, en l'emmenant à rencontrer des œuvres à la fois atypiques, mais aussi relativement proches de ses goûts.

Pour cela nous allons utiliser les caractéristiques des contenus (artistes, albums, labels), c'est-à-dire des métadonnées issues de l'analyse de texte (les données biographiques des artistes), des caractéristiques portant sur les genres attribués aux artistes. Nous récupérons pour cela ces données sur des sites possédant des API ouvertes telles que Blitzr, MusicBrainz pour la musique ou DBpedia pour les autres champs créatifs. L'entreprise française *Niland*, experte en analyse spectrale, fournissait à 1D Lab jusqu'en 2017 (date de son rachat par *Spotify*) un service de caractérisation des morceaux de musique (des tags déduits du son), et un service de recommandation basé sur ces tags.

Nous allons nous intéresser à trouver des moyens de description des œuvres numériques et d'utiliser ces caractéristiques pour définir des mesures de similarité et de dissimilarité pertinentes et qui utilisent des concepts compréhensibles par l'être humain. Une partie de notre travail sera aussi de trouver des moyens de calculer des similarités ou des dissimilarités dans les cas où les données à notre disposition sont très peu ou très mal définies.

À l'aide de ces caractéristiques, nous allons chercher à mettre en place des systèmes de recommandation qui apporteront de la nouveauté et de la diversité aux utilisateurs. Nous chercherons à être « audacieux » en partant des habitudes des utilisateurs et en adoptant deux stratégies. Une première consistera à apporter de la diversité, en regroupant les goûts des utilisateurs par catégories et en créant des listes diversifiées à partir de ces sous-catégories. Une deuxième consistera à rechercher les éléments les plus particuliers de sa liste d'éléments qu'il a aimés et d'apporter de l'audace dans les recommandations en sélectionnant pour l'utilisateur pour chaque élément qu'il a aimé un contenu non similaire, pas trop différent de ses goûts initiaux, mais dans une marge intermédiaire calculée entre ce qui est trop proche et ce qui est trop différent.

# Organisation du mémoire

Dans la première partie, nous ferons un état de l'art des systèmes de recommandation avec notamment le chapitre 1 où nous présentons les différentes méthodes de recommandation, le chapitre 2 où nous indiquons quelles sont les limites des systèmes de recommandation actuels et le chapitre 3 où nous parlons de diversité, nouveauté et sérendipité dans les systèmes de recommandation.

Dans la deuxième partie, nous présentons nos contributions, avec dans le chapitre 5 nos approches pour représenter les articles sous forme de vecteurs et nos méthodes de calcul de similarité importantes pour nos modèles de recommandations. Le chapitre 6 décrit nos approches de recommandation basées sur le contenu qui apportent de la diversité et de la nouveauté dans les listes de recommandations. Dans le chapitre 7, nous présentons des expériences qui valident nos approches.

Dans la dernière partie de la thèse, nous faisons une conclusion générale dans le chapitre 8 où nous rappelons la nécessité de mettre de la diversité et de la nouveauté dans les systèmes de recommandation. Dans le dernier chapitre de la thèse, le chapitre 9, nous présentons les travaux de recherche qui peuvent faire suite à cette thèse.

# Liste des publications

Les idées et les résultats présentés dans cette thèse font partie de divers articles de recherche évalués par des pairs. Dans cette section nous donnons la liste des publications.

## Conférence Internationale

— Lhérisson, P.-R., F. Muhlenbach, and P. Maret. Fair Recommendations Through Diversity Promotion. In G. Cong, W.-C. Peng, W. E. Zhang, C. Li, and A. Sun (Eds.), Advanced Data Mining and Applications – 13th International Conference, ADMA 2017, Singapore, November 5-6, 2017, Proceedings, Volume 10604 of Lecture Notes in Computer Science, pp. 89–103. Springer. 2017. Conference best paper award. doi : 10.1007/978-3-319-69179-4_7.

## Conférence Nationale

— Lhérisson, P.-R., F. Muhlenbach, and Pierre Maret. Recommandations et prédictions de préférences basées sur la combinaison de données sémantiques et de folksonomie. In F ; L. Gandon and G. Bisson (Eds.), 17ème Journées Francophones Extraction et Gestion des Connaissances, (EGC'2017) Actes, 24-27 Janvier 2017, Grenoble, France, Volume RNTI-E-33, Revue des Nouvelles Technologies de l'Information, pp. 333-338. URL `http://editions-rnti.fr/?inprocid=1002294`

— Lhérisson, P.-R., F. Muhlenbach, and Pierre Maret. Application mobile pour l'évaluation d'un algorithme de calcul de distance entre des items musicaux. In F. L. Gandon and G. Bisson (Eds.), 17ème Journées Francophones Extraction et Gestion des Connaissances, (EGC'2017) 24-27 Janvier 2017, Grenoble, France, Volume RNTI-E-33, Revue des Nouvelles Technologies de l'Information, S. 461–464. URL `http://editions-rnti.fr/?inprocid=1002324`

— Claquin, C., P.-R. Lhérisson. Au service de la création indépendante, Où va la musique ? Numérimorphose et nouvelles expériences d'écoute, Philippe Le Guern, pp.153-166. Presse des Mines, Libres opinions, Paris, France, 2016. `https://www.pressesdesmines.com/produit/ou-va-la-musique/`.

## Brevet National

— Muhlenbach, F., P.-R. Lhérisson, and P. Maret, (2017). Procédé de sélection automatique d'un contenu multimédia dans une base de données. Brevet d'invention FR3046269, INPI, Juin 2017

## Workshop International

— D. Diefenbach, Lhérisson, P.-R., F. Muhlenbach, and P. Maret. Computing the Semantic Relatedness of Music Genre using Semantic Wrb. In M. Martin, M. Cuquet and E. Folmer (Eds.), Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change and Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016. Bd. 1695, `CEUR-WS.org`, 2016. - URL `http://ceur-ws.org/Vol-1695/paper23.pdf`

# SOMMAIRE

Première partie

# État de l'art

Chapitre 1

# Anatomie d'un système de recommandation

## 1.1 Définition générale des systèmes de recommandations

Les systèmes de recommandation sont un ensemble de techniques et d'outils logiciels qui ont pour but de proposer à des utilisateurs des articles [Ricci *et al.*, 2015]. Ils sont implémentés sur des plateformes de diffusion de contenus multimédias (*Netflix*, *YouTube*, *Spotify*...), sur des plateformes de vente en ligne (*Zalando*, *Amazon*...), sur des réseaux sociaux (*Facebook*, *Twitter*...), etc. De ce fait, le terme « article » est un terme général pour nommer la ressource recommandée aux utilisateurs par ces systèmes, soit des films, des vidéos, des musiques, des vêtements, des livres, des utilisateurs, etc. Les systèmes de recommandation prennent tout leur sens quand les nombres d'utilisateurs et d'articles deviennent très importants. Car les utilisateurs ont peu de chances de connaitre toute la richesse du catalogue proposé par le service, et par ailleurs on peut faire valoir qu'il est quasiment impossible de faire une prescription humaine personnalisée pour tous les utilisateurs d'un service déployé et mis en production.

L'objectif du système de recommandation est de guider les utilisateurs dans les grandes masses de données disponibles, en particulier dans les plateformes de cybercommerce, de filtrer ces données pour proposer automatiquement à chaque consommateur les articles qui sont susceptibles de l'intéresser. Le processus de recommandation peut aussi être bénéfique pour le service, par exemple en créant de la demande (proposer d'autres articles pertinents), en créant de l'activité sur le site (par exemple, un utilisateur commence par regarder un tutoriel sur l'apprentissage profond *deep learning* et finit sur un documentaire sur le secret des commandos...), etc. La mise en place d'un système de recommandation peut servir plusieurs objectifs, que ce soit du point de vue du consommateur ou du fournisseur [Jannach et Adomavicius, 2016]. Formellement, la finalité du système est de maximiser pour chaque utilisateur de l'ensemble $U$ et chaque article recommandable de l'ensemble $I$ une fonction d'utilité $f : U \times I \rightarrow R$, qui permettra d'obtenir une liste de recommandations $R$. L'utilité souvent représentée par une valeur d'appréciation peut être n'importe quelle fonction (formule 1.1) qui permettra d'obtenir pour chaque utilisateur $u \in U$ l'article $i \in I$ qui sera ajouté à sa liste $R$ de recommandations.

$$\forall u \in U, i_u^* = argmax(f(u,i)), i \in I \tag{1.1}$$

L'utilité, outre les utilisateurs et les articles, est la troisième composante à laquelle il faut s'intéresser dans les systèmes de recommandations. L'utilité peut être définie comme une valeur

mesurant le degré de préférence d'un utilisateur pour un article. Cette valeur peut être induite à partir des données récupérées dans les sites de cybercommerce concernés. Dans des sites tels que *Amazon* ou sur l'*Apple Store*, ce degré de préférence est représenté par une certaine valeur entière (souvent de 1 à 5) qui représente l'attribution effectuée par un utilisateur pour un article donné. L'utilité peut être exprimée explicitement par l'utilisateur (étoiles, j'aime-*like*) d'autres fois elle l'est plutôt de manière implicite (p ex. le nombre de fois que l'utilisateur a écouté un genre musical, le nombre de fois qu'un utilisateur a visité la page d'un artiste, etc.). Ces informations sont souvent récupérées à l'insu de l'utilisateur. Les triplets de la forme utilisateur-article-préférence forment la matrice d'utilité. Seulement en règle générale, cette matrice est souvent vide et un grand nombre d'entrées restent inconnues. Une entrée vide signifie que l'utilisateur n'a pas interagi avec l'article en question. La prédiction de cette valeur d'utilité nous indiquera si cet article pourrait plaire et être recommandé à l'utilisateur.

Pour retrouver cette valeur d'utilité, les systèmes de recommandation adoptent différentes stratégies. Elles sont couramment classées en deux catégories :

— le filtrage collaboratif (*collaborative filtering*) qui se base sur la similarité entre les utilisateurs et/ou les articles. Les articles recommandés à un utilisateur seront ceux qui ont été appréciés par des utilisateurs aux goûts semblables aux siens ;

— le filtrage basé sur le contenu (*content-based*) qui utilise les caractéristiques des articles à recommander. Le système de recommandation va étudier par exemple, les genres des films, les réalisateurs, et les acteurs que préfère un utilisateur pour lui suggérer des films qui pourraient l'intéresser.

L'une des différences entre les deux méthodes est que la première se base sur toutes les interactions passées utilisateurs/articles (les usages des utilisateurs) pour produire des recommandations pour un utilisateur tandis que la deuxième ne se base que sur l'historique de l'utilisateur pour lui produire des recommandations. Dans la suite de ce chapitre, nous allons présenter plus en détail ces approches de recommandation. Nous allons parler du filtrage collaboratif dans la section 1.2.1 et du filtrage basé sur le contenu dans la section 1.2.2. Nous allons présenter finalement les méthodes hybrides de recommandations qui résultent d'une combinaison des deux méthodes citées préalablement dans la section 1.2.3.

## 1.2 Les différentes approches de recommandation

Pour retrouver les articles que l'utilisateur devrait apprécier, le système de recommandation doit prédire qu'un article vaut la peine d'être recommandé [Ricci *et al.*, 2015]. Le système prédit la valeur d'utilité pour des articles pas encore consultés par l'utilisateur, les compare, et établit une liste d'articles à recommander. Pour faire sa prédiction, le système va se baser sur les usages des utilisateurs (filtrage collaboratif) ou sur la description des articles (basé sur le contenu). Il existe des systèmes de recommandation qui utilisent les données contextuelles [Adomavicius et

Tuzhilin, 2015], telles que l'heure, la position, le jour de la semaine, etc. La formule d'utilité $f : U \times I \rightarrow R$ présenté dans définition générale (section 1.1) devient $f : U \times I \times C \rightarrow R$ avec $C$ pour « contexte ». Les recommandations peuvent aussi dépendre entre autres des données démographiques, de l'âge de l'utilisateur, du niveau de connaissance de l'utilisateur sur le domaine [Burke, 2007].

Il existe d'autres méthodes de recommandation à utiliser selon l'objectif, le domaine à traiter et les données disponibles. On retrouve les systèmes démographiques qui se basent sur le profil démographique d'un utilisateur (son âge, son sexe, son niveau d'étude, etc.) Les recommandations sont produites pour différents groupes de personnes en combinant leurs habitudes. Les informations démographiques ont été utilisées pour recommander des lieux touristiques à visiter dans certaines villes [Wang *et al.*, 2012]. On retrouve aussi les systèmes de recommandation basés sur la connaissance. Ce type de système de recommandation tente de suggérer des objets basés sur des inférences concernant les besoins et les préférences d'un utilisateur. Les recommandations basées sur les connaissances reposent sur des connaissances fonctionnelles : elles ont des connaissances sur la façon dont un élément particulier répond à un besoin particulier d'un utilisateur, et peuvent donc raisonner sur la relation entre un besoin et une recommandation possible [Burke, 2007]. Ces deux méthodes de recommandation sont des méthodes à part entière de recommandation, mais la frontière avec les méthodes plus classiques est très faible. Les données démographiques peuvent être utilisées comme un contexte, ou de concert avec les données comportementales des utilisateurs. Les systèmes de recommandation basés sur la connaissance peuvent être vus comme des sortes de systèmes de recommandation basés sur le contenu. Dans la suite de l'état de l'art, nous allons nous restreindre à la classification qu'on retrouve dans plusieurs études et présenter le filtrage collaboratif, les méthodes qui se basent sur le contenu et les méthodes hybrides [Adomavicius et Tuzhilin, 2005].

### 1.2.1 Le filtrage collaboratif

Le filtrage collaboratif utilise la matrice d'utilité pour déduire les articles à recommander. Cette méthode dépend de la communauté pour produire des recommandations. Dans sa version originale, la recommandation pour un utilisateur était faite en filtrant les articles selon des annotations laissées par d'autres utilisateurs [Goldberg *et al.*, 1992]. Ce système, appelé *Tapestry* permettait de filtrer les messages électroniques d'un utilisateur en fonction de l'usage des autres utilisateurs. *Tapestry* enregistrait les réactions des utilisateurs face à la masse de messages électroniques qu'ils recevaient. Le système avait accès aux courriers ouverts, aux courriers répondus et aux courriers supprimés. Il utilisait ces informations pour ranger la boîte de réception des utilisateurs. Les courriels ouverts, transférés, étaient placés en tête de liste, les courriels supprimés ou non ouverts étaient placés en bas de la liste. Depuis le filtrage collaboratif a évolué et l'idée telle qu'on la connait maintenant est apparue dans l'article de Resnick *et al.* [1994]. L'idée générale suit le principe suivant : des utilisateurs qui ont évalué des articles de

la même manière dans le passé auront beaucoup de chances de produire la même évaluation dans le futur [Breese *et al.*, 1998]. De même, un utilisateur évaluera deux articles de manière similaire, si d'autres utilisateurs ont donné les mêmes évaluations à ces deux articles [Deshpande et Karypis, 2004]. Le filtrage collaboratif n'utilise pas les descripteurs des articles, et fonctionne indépendamment de la nature des articles. Tout ce qui importe c'est la présence d'interactions (évaluations).

Les approches du filtrage collaboratif sont souvent divisées en deux groupes : les approches basées sur le voisinage (section 1.2.1.1), et les approches modèles (section 1.2.1.2) [Breese *et al.*, 1998].

#### 1.2.1.1 Les approches basées sur le voisinage

Les systèmes de recommandations basés sur le voisinage partent du principe que des gens qui ont les mêmes goûts aimeront les mêmes articles, de même que des articles similaires sont appréciés par des gens qui ont les mêmes goûts. On assemble ce qui se ressemble. Si vous êtes amateur de jeux de stratégie, vous aurez plus de chance de suivre les recommandations de votre ami amateur comme vous de jeux de stratégie, que celui qui joue aux jeux de tirs à la troisième personne (*third-person shooter*). Les approches basées sur le voisinage sont utilisées pour prédire le degré de préférence qu'un utilisateur aura pour des articles qu'il n'a pas encore consommés. Il existe deux manières de procéder, l'une dite basée sur l'utilisateur [Konstan *et al.*, 1997] et l'une dite basée sur l'article [Deshpande et Karypis, 2004].

**1.2.1.1.1 Notation** Nous allons utiliser la notation suivante dans les formules qui seront présentées dans la suite de cette section :

| Paramètres | Explication |
|---|---|
| u, v | utilisateur |
| i, j | article |
| $U_i$ | ensemble des utilisateurs qui ont apprécié un article $i$ |
| $I_u$ | ensemble des articles appréciés par un utilisateur $u$ |
| $r_u i$ | préférence d'utilisateur $u$ pour un item $i$ |
| $\overline{r}$ | valeur de préférence normalisée |
| $\hat{r}$ | valeur de préférence prédite |
| $N(u)$ | utilisateurs semblables à un utilisateur u |
| $w$ | valeur de pondération |

Tableau 1.1 – Notation et paramètres utilisés dans les mesures d'évaluation.

**1.2.1.1.2 Les méthodes basées sur l'utilisateur** Ces méthodes font la prédiction du degré de préférence d'un utilisateur pour un article ($r(u,i)$) en utilisant les préférences des utilisateurs similaires (les voisins).

La prédiction se fait en deux étapes : une première où l'on recherche les utilisateurs semblables, et une deuxième étape où l'on utilise les évaluations des utilisateurs similaires pour prédire les évaluations inconnues. La première étape repose sur un calcul de similarité. Il s'agit d'une étape importante qui peut impacter la précision et les performances du système de recommandation. Il existe dans la littérature plusieurs mesures populaires de similarité qui ont été utilisées dans le domaine de la recommandation : la similarité du cosinus (formule 1.2) [Billsus et Pazzani, 2000], la corrélation de Pearson (formule 1.3).

$$cos(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I_u} r_{ui}^2 \sum_{j \in I_v} r_{vj}^2}} \tag{1.2}$$

$$PC(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \overline{r}_u)(r_{vi} - \overline{r}_v)}{\sqrt{\sum_{i \in I_u v} (r_{ui} - \overline{r}_u)^2 \sum_{j \in I_u v} (r_{vj} - \overline{r}_v)^2}} \tag{1.3}$$

Il existe d'autres mesures de similarité souvent moins performantes que la corrélation de Pearson [Herlocker *et al.*, 2002] telles que la différence de la moyenne des carrés *Mean Squared Difference* [Shardanand et Maes, 1995].

L'étape de prédiction qui suit peut se faire soit en appliquant des méthodes statistiques de régression soit en faisant une classification. Pour la régression, la valeur prédite de préférence d'un utilisateur pour un article s'estime en faisant la moyenne des préférences données à cet article par les utilisateurs semblables (formule 1.4). Pour la classification, la valeur de préférence prédite est obtenue en faisant que les $k$ utilisateurs les plus semblables votent pour la valeur la plus probable. Cette préférence est tirée d'une liste de valeurs possibles.

$$\hat{r}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} r_{vi} \tag{1.4}$$

**1.2.1.1.3 Les méthodes basées sur l'article** Les méthodes basées sur l'article sont vues comme une transposition de la matrice d'utilité utilisateur × article. La prédiction de la valeur de préférence se fait en recherchant les articles similaires [Natarajan *et al.*, 2013]. Pour cette méthode, les mesures de similarité sont données dans la formule 1.5 pour la similarité du cosinus, dans la formule 1.6 pour la corrélation de Pearson et dans la formule 1.7 pour la valeur de préférence prédite.

$$cos(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} r_{uj}}{\sqrt{\sum_{u \in U_j} r_{ui}^2 \sum_{u \in U_j} r_{uj}^2}} \tag{1.5}$$

$$PC(i,j) = \frac{\sum_{u \in \mathcal{U}_{ij}} (r_{ui} - \overline{r}_i)(r_{uj} - \overline{r}_j)}{\sqrt{\sum_{u \in \mathcal{U}_u v} (r_{ui} - \overline{r}_i)^2 \sum_{u \in \mathcal{U}_i j} (r_{uj} - \overline{r}_j)^2}} \tag{1.6}$$

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u(i)} w_{ij} r_{uj}}{\sum_{j \in N_u(i)} |w_{ij}|} \tag{1.7}$$

Les méthodes basées sur le voisinage font parfois usage de techniques de normalisation pour éviter le biais observable dans les habitudes d'évaluation ($\overline{r}$) (centrée [Resnick *et al.*, 1994], centrée réduite [Herlocker *et al.*, 1999]). De même, la similarité entre articles ou entre utilisateurs est pondérée dans la formule finale de prédiction ($w$) [Herlocker *et al.*, 1999]. Les calculs de similarité ne prennent en compte que les articles coévalués, il faut réduire les effets indésirables liés au nombre d'articles coévalués [Ma *et al.*, 2007].

Pour conclure, les systèmes de recommandation basés sur le voisinage sont des méthodes pionnières des systèmes de recommandation. Elles produisent des prédictions relativement bonnes. Elles sont basées sur des mécanismes simples à implémenter. En revanche, elles nécessitent de charger en mémoire toute la base de données pour faire une prédiction (*memory-based*). Elles souffrent aussi du démarrage à froid et échouent parfois à produire des recommandations dans certains cas, par exemple pour un utilisateur qui ne partage aucune évaluation en commun avec les autres utilisateurs [Ricci *et al.*, 2015].

#### 1.2.1.2 Les approches modèles

**1.2.1.2.1 Notation** Nous allons utiliser la notation suivante dans les formules qui seront présentées dans la suite de cette section, elle complète la notation du tableau 1.1

| Paramètres | Explication |
|---|---|
| $p_u$ | vecteur associé à un utilisateur $u$ |
| $q_i$ | vecteur associé à un item $i$ |
| $f$ | rang des vecteurs associés |
| $\mu$ | biais général sur les évaluations |
| $b_u$ | biais pour un utilisateur |
| $b_i$ | biais pour un utilisateur |
| $w$ | valeur de pondération |

Tableau 1.2 – Notation et paramètres utilisés dans les mesures d'évaluation.

À la suite de la compétition lancée par *Netflix* en octobre 2006 (*The Netflix Prize competition*), on a pu observer des avancées dans le domaine des systèmes de recommandation basés sur le filtrage collaboratif. Les méthodes qui s'inspirent de la décomposition de matrice ont gagné en popularité grâce à cette compétition [Koren *et al.*, 2009; Salakhutdinov et Mnih, 2008]. Les approches reposant sur des modèles sont sorties victorieuses et il en a découlé l'implémentation dans des services (*Spark* [1]) de ces algorithmes [Zhou *et al.*, 2008].

Les systèmes de recommandation basés sur des modèles impliquent la construction d'un modèle en se servant de l'ensemble des évaluations. En d'autres termes, ces algorithmes extraient de l'information des données et l'utilisent comme un modèle pour calculer des recommandations sans avoir à réutiliser l'ensemble de données à chaque fois. Ces méthodes ont l'avantage d'être rapides lors de la phase de prédiction, et sont plus précises. Dans cette famille de modèles, on

1. `https://spark.apache.org/docs/latest/mllib-collaborative-filtering.html`

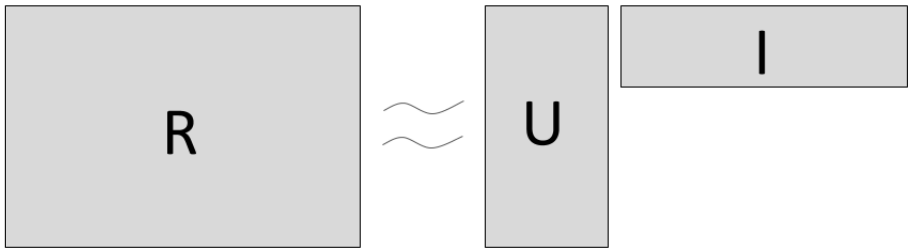



FIGURE 1.1 – Décomposition d'une matrice en facteurs latents.

retrouve les modèles de facteurs latents, comme la factorisation de matrice (Décomposition en valeurs singulières, SVD) qui place les articles et les utilisateurs dans le même espace vectoriel. L'usage de la décomposition en valeurs singulières dans les systèmes de recommandation a nécessité des adaptations de l'algorithme initial. En effet, la particularité d'avoir une matrice creuse et de rang élevé ne constitue pas des conditions à priori pour que le SVD fonctionne. Kim et Yum [2005] ont proposé de remplacer les valeurs manquantes de la matrice d'utilité par la moyenne des évaluations. Mais la matrice devenait dense, le calcul couteux, et la méthode ne produisait pas des résultats probants. Des approches ont été proposées pour travailler avec les matrices vides caractéristiques des systèmes de recommandation. Ces approches dépendent de paramètres de régularisation qui empêchent l'algorithme de surapprendre [Koren, 2008]. Ces approches ne sont pas à proprement parlé des applications mathématiques de la décomposition de matrice comme dans l'article de Billsus et Pazzani [1998], mais plutôt une manière de calculer une version approximative de la matrice de rang réduite en minimisant l'erreur quadratique (nous allons quand même utiliser le terme *SVD* adopté par la communauté pour la nommer), figure 1.1. La version initiale de cet algorithme a été présentée par Simon Funk dans un article de son blog *Netflix Update: Try This at Home* [2]. Il s'agit d'une méthode qui vise à décomposer la matrice vide d'utilité en deux matrices de rang réduit qui représentent les utilisateurs ($p \in \mathbb{R}^f$) et les articles ($q \in \mathbb{R}^f$). Les utilisateurs et les articles sont donc placés dans un même espace vectoriel latent de dimension $f$. Chaque utilisateur $u$ est associé à un vecteur $p_u \in \mathbb{R}^f$ et chaque article $i$ est associé à un vecteur $q_i \in \mathbb{R}^f$. Cette factorisation de matrice se fait en utilisant une approche itérative visant à minimiser l'erreur d'une fonction objectif. L'algorithme du gradient stochastique est souvent utilisé. La fonction de perte (*Loss Function*) à minimiser inclut en règle générale, un

2. http://sifter.org/~simon/journal/20061211.html

biais général ($\mu$) sur les évaluations, un biais pour l'utilisateur ($b_u$) sur ses habitudes et un autre pour l'article ($b_i$) sur la popularité, formule 1.8.

$$min_{p,q,b} \sum_{ui} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(||p_u||^2 + ||q_i||^2 + b_u^2 + b_i^2) \tag{1.8}$$

Cette formule permet de retrouver les vecteurs utilisateurs et articles de rang $f$ qui permettront par la suite de prédire les valeurs de préférence et compléter la matrice d'utilité. Il suffit de faire le produit scalaire des deux vecteurs et d'ajouter le biais, formule 1.9 :

$$\hat{r} = b_{ui} + p_u^T q_i \tag{1.9}$$

D'autres modèles de factorisation de matrice [Takács *et al.*, 2008] emploient une forme de factorisation dite probabiliste (*Probabilistic matrix factorization*), et arrivent à passer à l'échelle sur de grands jeux de données tout en produisant de meilleurs résultats que les techniques *SVD* standards.

Pour améliorer les techniques *SVD*, Yehuda Koren a proposé d'inclure dans la formule originale (formule 1.8), des informations implicites [Koren, 2008]. Dans cette version appelée couramment *SVD++*, Koren ajoute à $p_u$ un facteur $|N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_i$, qui permet de prendre en compte l'action d'évaluation de l'utilisateur (formule 1.10). En effet la probabilité qu'un utilisateur apprécie un article qu'il a pris le temps d'évaluer est plus élevée que la probabilité qu'il apprécie un article pris aléatoirement dans l'ensemble de ceux qu'il n'a pas évalués. Cette méthode *SVD++* donne de meilleurs résultats que la méthode *SVD*.

$$min_{p,q,b} \sum_{ui} (r_{ui} - \mu - b_u - b_i - q_i^T (p_u|^{\frac{-1}{2}} \sum_{j \in N(u)} y_i))^2 + \lambda(||p_u||^2 + ||q_i||^2 + b_u^2 + b_i^2 + \sum_{j \in N(u)} ||y_i||^2) \tag{1.10}$$

Le modèle *SVD++* a été étendu pour qu'il prenne en compte l'aspect temporel dans les recommandations (*TimeSVD++*) en modélisant différemment le vecteur utilisateur Koren [2010](formule 1.11).

$$p_{uk}(t) = p_{uk} + \alpha_{uk} \cdot dev_u(t) + p_{uk,t} k = 1, ..., f \tag{1.11}$$

Les préférences d'un utilisateur varient avec le temps, de même que les interactions observées entre les utilisateurs et les articles. Certains travaux ont cherché à ajouter directement une dimension temporelle à la matrice d'utilité pour créer un tenseur [Xiong *et al.*, 2010]. L'aspect temporel a aussi été pris en compte en ajoutant une fonction qui atténuait ($f(x) = e^{-\infty t}$) l'effet des anciennes évaluations lors du calcul de similarité et lors de la prédiction [Liu *et al.*, 2010]. Ces approches ont montré de bons résultats en améliorant l'erreur quadratique liée à la prédiction [Ricci *et al.*, 2015]. Ces méthodes de recommandation prennent en compte le contexte (*context-aware recommendations*).

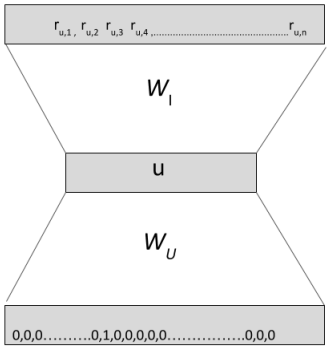



FIGURE 1.2 – Représentation sous forme de réseau de neurone de la décomposition de matrice.

Les méthodes de prédiction de préférence ont donné lieu à la publication d'un article qui relate l'utilisation d'une technique d'apprentissage profond pour résoudre le problème de complétion de la matrice de préférence [Salakhutdinov *et al.*, 2007]. Cette méthode a été testée sur le jeu de données du *Netflix prize* et a permis d'améliorer de 6% le taux d'erreur du système de *Netflix*. Cet article est considéré comme la première application réussie des réseaux de neurones dans les systèmes de recommandation.

Les systèmes de factorisation de matrice peuvent être considérés implicitement comme des systèmes d'apprentissage profond. Ils peuvent être vus comme une approche simpliste des réseaux de neurones (figure 1.2).

— en **entrée**, le réseau prend le vecteur binaire (*One-hot*) de l'identifiant de l'utilisateur (tous les utilisateurs représentés par un 0 sauf l'utilisateur ciblé représenté par un 1) ;

— **les poids cachés connectés au vecteur en entrée** sont la matrice des caractéristiques de l'utilisateur ;

— **la couche cachée** est représentée par le vecteur latent de l'utilisateur ;

— **les poids cachés connectés au vecteur en sortie** sont la matrice des caractéristiques des articles ;

— en **sortie**, les préférences de l'utilisateur pour tous les articles.

Mais l'apprentissage profond [LeCun *et al.*, 2015] a surtout eu du succès dans la communauté à partir des publications sur *Word2Vec* de Mikolov *et al.* [2013]. Le *Word2Vec* utilise des réseaux de neurones peu profonds pour apprendre des représentations vectorielles des mots. Le but est d'apprendre les vecteurs des mots en fonction du contexte *CBOW* ou le contraire, le contexte en fonction d'un mot *Skip-Gram*. Le contexte est souvent composé des 1 à 5 mots qui entourent un mot dans un corpus de texte. *Word2Vec* a été étendu par la suite à *Paragraph2vec, doc2vec* [Le et Mikolov, 2014]. Ce modèle permet d'apprendre la représentation d'un paragraphe ou d'un document. Le plongement lexical du paragraphe ou du document permet d'ajouter un contexte global au modèle. Ces modèles ont inspiré les modèles dits « -2vec » qui apprennent

un plongement pour les articles. Les méthodes *Prod2vec* de Grbovic *et al.* [2015] utilisent le modèle *Skip-gram* sur les articles. En entrée, pour chaque utilisateur, il prend le énième article acheté par l'utilisateur et utilise comme contexte les autres articles achetés par l'utilisateur. Il permet d'apprendre un vecteur de représentation pour chaque article. Leur modèle permet aussi de représenter des ensembles d'articles. En entrée, il prend les articles achetés par un utilisateur dans un panier, et le contexte est représenté par les autres paniers des utilisateurs. Pour apprendre un vecteur de représentation des utilisateurs, ils s'inspirent de *Paragraph2vec*. Le plongement de l'utilisateur ajoute un contexte global. En entrée, le modèle prend l'utilisateur, et les articles achetés sauf le énième article. La sortie est le énième article acheté par l'utilisateur. D'autres modèles similaires ont été proposés comme le *item2vec* de Barkan et Koenigstein [2016] qui utilise aussi le modèle *Skip-Gram* avec un échantillonnage négatif.

Les autoencodeurs utilisés en premier dans les systèmes de recommandation [Salakhutdinov *et al.*, 2007] permettent de reconstruire en sortie des données qui ont été corrompues en entrée (ajout de 0 aléatoirement dans le vecteur d'entrée) [Vincent *et al.*, 2008]. Ils ont été utilisés par la suite par Wang *et al.* [2015] et Wu *et al.* [2016] pour reconstruire les vecteurs corrompus d'interaction. Une version utilisant un réseau de neurones récurrent a été proposée par la suite [Wang *et al.*, 2016].

Pour conclure, les méthodes collaboratives à base de modèles arrivent à résoudre certains problèmes rencontrés par les modèles basés sur le voisinage. Le passage à l'échelle qui était problématique semble être résolu, les modèles créés par ces méthodes sont plus petits que la base de données et peuvent être utilisés efficacement. Les modèles profonds gardent aussi ce facteur important à l'esprit, de plus ils sont implémentés avec succès dans des systèmes de recommandation dans l'industrie (*Google* [Cheng *et al.*, 2016], *Youtube* [Covington *et al.*, 2016]). L'apprentissage de ces modèles peut se faire hors-ligne et être mis en ligne dans une application qui produira des prédictions pour chaque utilisateur sans qu'il soit nécessaire de requêter toute la matrice d'utilité. Pour la factorisation de matrice, l'utilisation des variables de biais permet de prévenir le surapprentissage, sans que la précision du modèle en pâtisse. Mais il faut noter que ces méthodes sont assez rigides, et qu'il est très difficile d'ajouter de nouvelles données à ces modèles. De surcroît il est difficile d'expliquer les recommandations aux utilisateurs, contrairement aux recommandations issues des méthodes de voisinage (même si elles manquent de transparence).

### 1.2.2 Le filtrage basé sur le contenu

Le filtrage basé sur le contenu permet de recommander les articles en établissant une comparaison entre le contenu des articles et un profil d'utilisateur [Lops *et al.*, 2011]. Les contenus sont le plus souvent représentés par un ensemble de termes issus des descriptifs des articles [Seroussi, 2010], et les profils des utilisateurs sont souvent représentés par ces mêmes termes. Le profil d'un utilisateur est constitué à partir des articles pour lesquels il a montré un intérêt (évalué,

consulté, aimé, etc.). Les systèmes de recommandation basés sur le contenu construisent donc un modèle pour chaque utilisateur.

Le filtrage basé sur le contenu est la méthode la plus populaire pour recommander des articles caractérisés par du contenu textuel [Beel *et al.*, 2016], par exemple les revues et les articles scientifiques. Le domaine de recherche des systèmes basés sur les contenus est très proche de la Recherche d'Information [Baeza-Yates et Ribeiro-Neto, 1999]. En effet ces systèmes s'appuient sur un modèle d'espace vectoriel, les profils des utilisateurs et des articles étant représentés par des vecteurs de termes pondérés de type *TF-IDF* [Salton, 1989]. Ces systèmes ont pour limite qu'ils n'arrivent pas à percevoir le sens des textes, ce qui crée une perte de l'information sémantique liée au profil de l'utilisateur et de l'article. Les technologies liées au web sémantique (ontologie, thésaurus, base de connaissance) sont des solutions qui ont été proposées pour répondre à cette problématique. Dans la suite de cette section nous allons présenter, les modèles vectoriels et les modèles sémantiques (section 1.2.2.2).

#### 1.2.2.1 Les modèles vectoriels

Les modèles vectoriels et une de leurs extensions, l'analyse sémantique latente, sont des méthodes qui utilisent des termes pour représenter les documents sous forme vectorielle dans des espaces à plusieurs dimensions. Ils ont été utilisés dans les systèmes de recommandation *Web*, *Letizia* [Lieberman, 1995], *Syskill & Webert* [Pazzani *et al.*, 1996], *Amalthea* [Moukas, 1997], *WebMate* [Chen et Sycara, 1998]. Ces systèmes s'appuient sur des retours implicites des utilisateurs pour comprendre leurs préférences, pages visitées, pages mis en favori ou sur des retours explicites des utilisateurs avec des systèmes « j'aime », « je n'aime pas ». Les documents et les profils des utilisateurs sont représentés par $n$ mots-clés qui permettront par la suite de produire des recommandations en comparant ces deux familles de vecteurs. Ces techniques ont aussi été employées dans la recommandation de textes liés à l'actualité [Billsus et Pazzani, 2000, 1999]. Au-delà du texte, certains systèmes de recommandation sur le contenu utilisent des « tags » pour qualifier les articles et les utilisateurs [Cantador *et al.*, 2010]. Les méthodes de pondération *BM25* [Robertson et Jones, 1976] ont servi à pondérer les vecteurs obtenus à partir des tags. Depuis l'augmentation exponentielle du nombre de sources textuelles sur le *Web*, plus particulièrement avec l'arrivée des bases encyclopédiques (*Wikipedia*, *Freebase*), les nouvelles techniques de représentation vectorielles liées au plongement lexical (*Word2vec*) [Mikolov *et al.*, 2013], ont aussi été utilisées pour représenter les articles et les utilisateurs [Musto *et al.*, 2015]. Ces techniques ont montré des résultats encourageants et ont laissé présager de futures exploitations pour les systèmes de recommandation basés sur le contenu.

L'usage de l'apprentissage profond est d'autant plus intéressant qu'il est capable d'extraire directement des caractéristiques des articles [LeCun *et al.*, 2015]. L'apprentissage profond permet au système de recommandation de travailler directement avec la donnée. Ceci est très utile quand la donnée peut être caractérisée par du son, des images, des vidéos, en plus du texte. Les réseaux neuronaux convolutifs ont été utilisés pour extraire des caractéristiques visuelles d'images pour

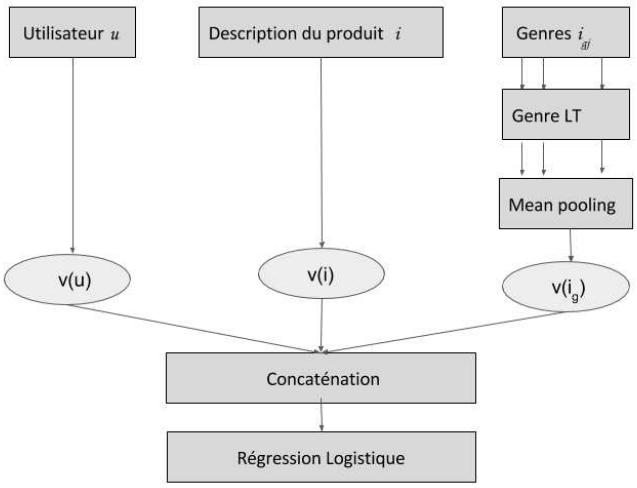



FIGURE 1.3 – Architecture d'un système de recommandation basé sur le contenu utilisant l'apprentissage profond Suglia *et al.* [2017].

apprendre une distance entre des articles [McAuley *et al.*, 2015]. La distance était personnalisée en fonction d'un utilisateur, et le but était de maximiser la probabilité qu'il existe une relation entre un utilisateur et un article. Un réseau de neurones convolutif a aussi été appliqué aux recommandations musicales [van den Oord *et al.*, 2013]. Le but ici était de résoudre le problème du démarrage à froid, c'est-à-dire produire des recommandations quand il y a peu de données d'écoute, réussir à recommander les musiques peu populaires ou nouvelles. Mais il est connu qu'il existe un écart sémantique entre le signal audio et les préférences d'un utilisateur. Ce problème peut être résolu en utilisant des données textuelles comme la biographie d'un artiste pour entrainer le réseau de neurones [Oramas *et al.*, 2017]. Les informations textuelles ont même la faculté d'améliorer les recommandations [Bansal *et al.*, 2016]. Une architecture a été proposée pour prédire la probabilité qu'un utilisateur apprécie un article [Musto *et al.*, 2016]. Ce travail utilise une forme de réseau de neurones récurrents, *Long short-term memory (LSTM)*. Il est composé de deux modules qui génèrent deux types de plongement, un pour les utilisateurs et un pour les articles. Le plongement pour les articles est appris à partir du texte de description fourni avec l'article. Cette architecture a été étendue en ajoutant différents modules qui prennent en compte d'autres types d'informations fournies avec les articles (p.ex les genres) [Suglia *et al.*, 2017], elle est schématisée à la figure 1.3. Le modèle présenté donne de meilleurs résultats que les systèmes de recommandation basés sur le filtrage collaboratif et les méthodes de plongement de mots sur les jeux de données *Movie Lens* et *DBbook*.

#### 1.2.2.2 Les modèles sémantiques

Les modèles sémantiques ont été introduits pour répondre aux problèmes rencontrés par les modèles naïfs basés sur le contenu. Le système de recommandation basé sur le contenu suggère

à un utilisateur des articles similaires à ceux qu'il a aimés dans le passé. La recommandation est générée en faisant correspondre la description des articles aux descriptifs enregistrés dans le profil d'un utilisateur [Lops *et al.*, 2011; Pazzani et Billsus, 2007]. Ces systèmes basés sur un modèle naïf utilisent le texte descriptif et sont peu efficaces pour traiter les polysèmes (Turkey, Apple...), les concepts similaires, les concepts qui peuvent être présentés par plusieurs mots (IA, intelligence artificielle). Ils ne reconnaissent pas non plus les entités nommées. Les modèles sémantiques ont pour but d'améliorer la représentation du contenu, d'être robustes face aux différents problèmes liés au traitement du langage naturel, de modéliser les préférences d'un utilisateur de manière efficiente [Gemmis *et al.*, 2015a]. Pour répondre à ce problème, il a été proposé de représenter les profils des utilisateurs et des articles par des concepts sémantiques. Une solution est d'utiliser des ontologies linguistiques pour la désambigüisation du sens des mots [Semeraro *et al.*, 2007]. Ici le sens d'un mot dans un texte est choisi dans un dictionnaire (p.ex *Wordnet* [3]) en prenant en compte le contexte dans lequel il apparait [Basile *et al.*, 2007]. Cette méthode permet de prendre en compte la polysémie et les synonymes. Pour reconnaître les entités nommées, il existe des systèmes dans l'état de l'art tels que *Open Calais* [4], *Tag.me* [5], *babelfy* [6]. La recherche d'entités nommées et les méthodes de désambigüisation du sens des mots utilisés pour augmenter les profils ont donné de meilleurs résultats en termes de précisions que les modèles basés uniquement sur le texte [Gemmis *et al.*, 2008].

Les modèles sémantiques font correspondre les descripteurs des articles à recommander à des concepts du *Web* sémantique. Cette expression a été inventée par Tim Berners-Lee et selon lui, le *Web* sémantique fournit un modèle qui permet aux données d'être partagées et réutilisées entre plusieurs applications, entreprises et groupes d'utilisateurs [Berners-Lee *et al.*, 2001]. Le *Web* sémantique prend forme dans le projet de données ouvertes liées (DOL, *Linked Open Data* en anglais). L'ambition de ce projet est de structurer et de connecter toutes les données disponibles sur le *Web*. Il existe un ensemble de bonnes pratiques à adopter pour publier des DOL sur le *Web* [Villazón-Terrazas *et al.*, 2011]. Les DOL utilisent les triplets RDF (Resource Description Framework) pour modéliser l'information et la rendre publique sur le *Web*. Les triplets RDF sont sous la forme Sujet - Prédicat - Objet, p.ex. on retrouve sur *DBPedia* le triplet « dbr :Keanu_Reeves » - « dbo :starring » - « dbr :The_Matrix » pour représenter l'information suivante : *Keanu Reeves* joue dans *The Matrix*. Le langage de requête SPARQL (SPARQL Protocol and RDF Query Language) est utilisé pour récupérer ces données. Les données liées sont un ensemble de jeux de données sémantiques interconnectés qui comporte 149 milliards de triplets et 9 960 jeux de données [7] (figure 1.4).

Les jeux de données du *Web* sémantique ont été utilisés pour améliorer les recommandations. Dans les cas où les données sont très peu caractérisées, les DOL permettent d'enrichir les

3. https://wordnet.princeton.edu
4. http://www.opencalais.com/opencalais-api/
5. https://tagme.d4science.org/tagme/
6. http://babelfy.org/
7. http://stats.lod2.eu

représentations. La découverte de liens entre les données peut donner lieu à des recommandations surprenantes, différentes de ce qu'un système naïf basé sur le contenu propose. Noia *et al.* [2012] ont utilisé par exemple *DBPedia* pour faire de la recommandation de films. Dans cet article, les informations récupérées du graphe de *DBPedia* ont été transformées pour être mises dans un tenseur, et l'indice de Jaccard est utilisé pour estimer une similarité entre les entités de la base de connaissance. La prédiction de la préférence d'un utilisateur pour un article a été calculée à l'aide de l'algorithme des *k* plus proches voisins. Cette approche exploitant les données sémantiques utilise aussi des représentations vectorielles. D'autres approches ont exploité directement le graphe, par exemple *DBrec*, un moteur de recommandation exploitant *DBPedia* pour faire des recommandations d'artiste [Passant, 2010]. On note que cette approche ne se base que sur la présence des liens entre les concepts. La sémantique des relations n'est pas exploitée, tous les liens sont traités avec la même importance sans prendre en compte les niveaux hiérarchiques exprimés dans *DBPedia*. D'autres approches ont utilisé des techniques de fouilles de graphes afin d'identifier les nœuds les plus pertinents dans le graphe [Musto *et al.*, 2017]. En effet même si les résultats obtenus par Musto *et al.* [2017] sont encourageants, il subsiste des problèmes compliqués à résoudre. En effet, les entités sont décrites par plusieurs propriétés qui ne sont pas toutes pertinentes. Une réduction de dimensionnalité s'impose et la solution passe par l'exploitation des caractéristiques du graphe (centralité, degré, etc.). Les techniques basées sur les graphes et les représentations utilisant les DOL ont permis d'améliorer les résultats des recommandations. Ceci a été appliqué aux recommandations de livres et de films [Musto *et al.*, 2017].

Somme toute, les systèmes de recommandation basés sur le contenu ont l'avantage de ne pas dépendre de la communauté pour construire le profil d'un utilisateur. Comme pour les systèmes basés sur le voisinage, il est facile d'apporter une explication aux recommandations (vous écoutez Kenny Arkana, vous allez apprécier Demi Portion) [Lops *et al.*, 2011]. Puisque ces systèmes ne dépendent pas des appréciations des autres utilisateurs pour produire des recommandations, un nouvel article non évalué, ou non consommé, peut se retrouver dans des listes de recommandations [Lops *et al.*, 2011]. Ces systèmes ne se basent donc pas à priori sur la popularité. En revanche, la pertinence des recommandations basées sur le contenu est dépendant de la qualité de la description des articles [Lops *et al.*, 2011]. Il est souvent difficile, mais pas impossible (*The Music Genome Project* [8]) de produire une caractérisation détaillée des articles [Lops *et al.*, 2011].

Comme nous l'avons vu pour les articles, le système est efficace seulement si le profil de l'utilisateur est bien décrit. Une bonne description des profils d'utilisateurs dépend d'une bonne description des articles et de la présence d'un minimum de données historiques. Les techniques d'apprentissage profond ont montré qu'elles pouvaient être utilisées dans les systèmes de recommandation basés sur le contenu. Pour tirer avantage de ces technologies, il faut avoir accès à un volume de données conséquent et à une bonne puissance de calcul pour pouvoir traiter

8. https://www.pandora.com/about/mgp

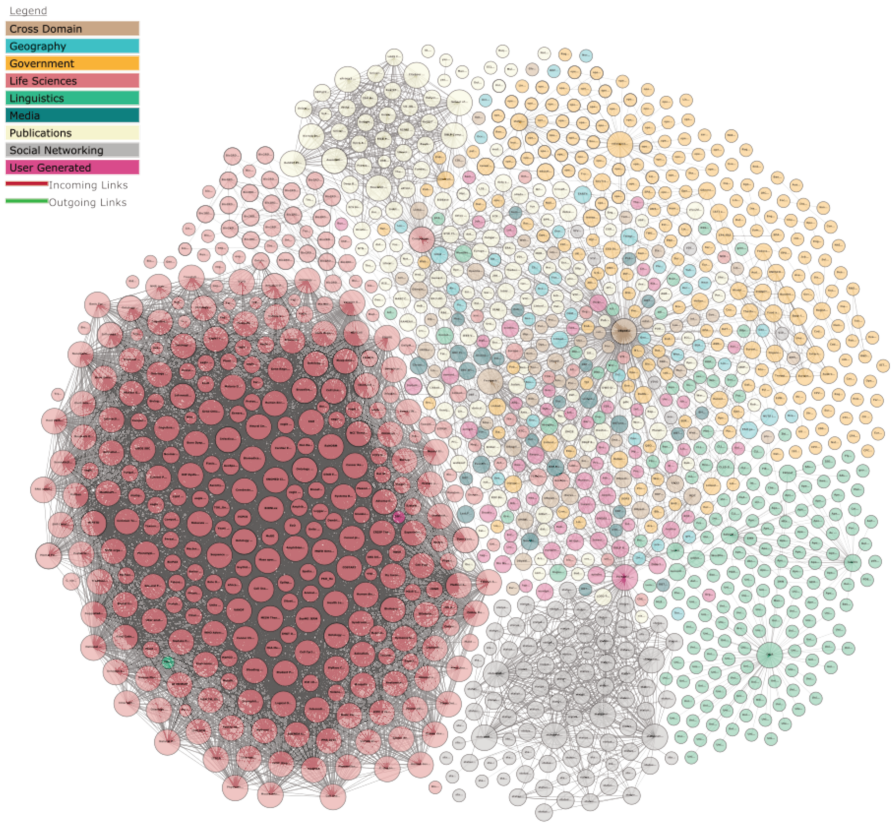



FIGURE 1.4 – Nuage des données liées ouvertes par domaines d'applications. Chaque disque représente un jeu de données et chaque arête un ensemble de liens entre les jeux de données (mise à jour le 22-08-2017)

ces données. Ces systèmes requièrent beaucoup de temps pour faire l'apprentissage. Mais dans un scénario réel, ces modèles peuvent être appris hors-ligne puis utilisés en ligne pour faire de la prédiction. Les systèmes de recommandation basés sur le contenu sont connus pour faire du surapprentissage, en conséquence ils fournissent des recommandations trop similaires à ce que l'utilisateur avait déjà consommé. En revanche, les modèles basés sur le *Web* sémantique peuvent, en exploitant le graphe, découvrir des liens inédits entre des entités et produire des recommandations moins similaires à ce que l'utilisateur avait déjà consommé.

### 1.2.3 Les méthodes hybrides

Les recommandations hybrides ont pour but de faire de meilleures prédictions en combinant les méthodes présentées au préalable (les méthodes basées sur le filtrage collaboratif, et les méthodes basées sur le contenu). Généralement, les systèmes de recommandations utilisent plusieurs techniques et il existe différentes manières de combiner ces différentes approches [Burke, 2002].

— **une méthode de pondération.** C'est une méthode qui combine les résultats de deux

approches en faisant par exemple une combinaison linéaire des scores de chaque technique de recommandation ;

— **une méthode de commutation.** Le système utilise un critère qui lui permet de changer entre les différentes techniques. Un scénario pourrait être que le système utilise une technique et si les résultats ne sont pas assez pertinents, il change et utilise une autre technique en espérant améliorer les résultats ;

— **une méthode mixte.** Dans cette approche, le système de recommandation étend les descriptions des jeux de données en utilisant les appréciations des utilisateurs ainsi que les descriptions des articles. La fonction de prédiction doit prendre en compte les deux types de données ;

— **technique en cascade.** La technique de la cascade est un processus étape par étape. Une première méthode de recommandation est appliquée qui produit un classement puis une seconde est appliquée pour améliorer le classement réalisé par la première méthode.

Dans l'apprentissage profond, diverses solutions se présentent pour faire des systèmes hybrides :

— **l'initialisation** consiste à obtenir des métadonnées pour représenter l'article et utiliser ces métadonnées comme vecteurs d'entrées dans le réseau de neurones ;

— **la régularisation** consiste à obtenir des métadonnées pour faire une représentation de l'article à recommander, puis obtenir une deuxième représentation de l'article proche de la première à partir des interactions, ajouter un terme de régularisation à la perte de cette différence ;

— **Le raccordement** consiste à concaténer la représentation à partir des métadonnées à celle apprise à partir des interactions pour obtenir un seul vecteur de représentation de l'article.

Les méthodes hybrides peuvent tirer le meilleur de deux techniques de recommandation, ou de plusieurs techniques pour créer un système de recommandation plus performant, plus robuste que les autres méthodes prises individuellement.

## 1.3 Conclusion

Dans ce chapitre, nous avons présenté les différentes techniques employées pour faire des recommandations. Nous avons dit que les différentes composantes d'un système de recommandation étaient les utilisateurs et les articles. Le but était de proposer à chaque usager du système de nouveaux articles en se basant sur son historique de consommation passé ou sur celui de la foule. Plusieurs techniques existent et peuvent être classées en trois grandes catégories : le filtrage collaboratif, le filtrage basé sur le contenu et une méthode hybride qui combine les deux premières méthodes.

Dans la catégorie du filtrage collaboratif, nous avons vu qu'il existe deux approches : les approches basées sur le voisinage et les approches modèles. Les approches basées sur le voisinage

ont l'avantage d'être efficaces et faciles à implémenter. Les approches basées sur le modèle peuvent être subdivisées en deux familles : celles basées sur l'utilisateur (*user-based*) et celles basées sur l'article (*item-based*). Les approches basées sur l'article présentent l'avantage d'être stables, ne nécessitant pas une mise à jour régulière des similarités, contrairement à l'approche utilisateur. Mais ces deux approches souffrent toutes les deux du démarrage à froid (il s'agit de la difficulté de faire une recommandation à un nouvel utilisateur ou de recommander un nouveau produit) et du peu d'interactions entre les utilisateurs et les articles produisant des matrices vides. Il est aussi très difficile de faire passer à l'échelle ces deux approches.

Les approches modèles utilisent des techniques de factorisation de matrice, des techniques de clustering, des réseaux bayésiens probabilistiques, des machines restreintes de Boltzmann, etc., pour apprendre des modèles à partir des données d'interactions du système. Nous avons vu que ces méthodes ont de nombreux avantages tels que, réduire la dimension des données tout en gagnant en précision, passer à l'échelle, être plus rapides que les approches basées sur le voisinage pour faire la prédiction. Mais ces approches modèles présentent le désavantage de manquer de flexibilité. En effet, il est difficile d'ajouter de nouvelles données au modèle, cela implique un réapprentissage du modèle, et cette tâche peut se révéler gourmande en puissance de calcul et en temps.

Pour les systèmes basés sur le contenu, nous avons divisé cette catégorie en deux : une première qui englobe les approches vectorielles, et une seconde qui englobe les approches sémantiques. Nous avons vu que les systèmes de recommandations basés sur le contenu ne souffrait pas de démarrage à froid ou du problème des matrices vides, car pour faire une recommandation à un utilisateur, il n'a pas besoin des historiques de consommation des autres utilisateurs. Nous avons vu que cette approche arrive à faire des recommandations à des utilisateurs qui ont des goûts particuliers et arrive à recommander aux utilisateurs des articles nouveaux et peu populaires. En utilisant cette approche, nous pouvons aussi expliquer aux utilisateurs pourquoi une recommandation leur est faite. Mais en revanche, il est difficile de trouver les caractéristiques appropriées pour certaines données, par exemple, la musique, les images, les vidéos. Ce genre de méthode n'arrive pas à prendre en compte les intérêts multiples qu'un utilisateur peut avoir, et ne recommande pas d'articles en dehors du profil de l'utilisateur.

Nous avons vu qu'il existe des méthodes hybrides, qui consistent à combiner des méthodes pour créer un système de recommandation. Le but est d'utiliser ce que les méthodes offrent de mieux pour pallier les problèmes qu'ils rencontrent quand ils sont utilisés seuls.

Nous analyserons dans le chapitre 2 la problématique de recommandation, les différents facteurs qui existent et qui peuvent avoir un effet sur la recommandation.

CHAPITRE 2

# QU'EST-CE QU'UNE « BONNE » RECOMMANDATION ?

Les systèmes de recommandation ont pour but de proposer à des utilisateurs des articles susceptibles de les intéresser. Il existe plusieurs moyens permettant de mesurer la pertinence des recommandations. Pour mesurer les performances des systèmes de recommandation, dans la littérature on retrouve souvent des évaluations hors lignes. Ces évaluations sont indépendantes des facteurs qui peuvent affecter une recommandation qui a lieu dans un système en production et se proposent de mesurer de manière objective les recommandations. Ces évaluations sont effectuées en fonction du problème de recommandation à résoudre : la prédiction des évaluations ou le classement des recommandations (top-$k$) [Steck, 2013]. Les évaluations hors ligne sont effectuées sur des jeux de données publiques. Les plus populaires restent ceux de *GroupLens* [1], avec les fameux jeux de données de *MovieLens* appelées 100k, 1M, 10M, 20M qui comme leur nom l'indique contiennent 100 mille, 1, 10 ou 20 millions d'évaluations et aussi des *tags* et des genres pour décrire les films. *GroupLens* est un groupe de recherche en informatique de l'Université de Minnesota spécialisé dans les systèmes de recommandation. Outre les jeux de données sur les films, *GroupLens* a aussi publié un jeu de données *Last.fm*. Òscar Celma, actuel directeur de la webradio *Pandora*, a aussi collecté deux jeux de donnés de *Last.fm* [2] et le groupe Labrosa de l'Université de Columbia a ajouté des données utilisateurs à son jeu de données *Million Song Dataset* [3] [McFee *et al.*, 2012].

Contrairement aux jeux de données *MovieLens*, les jeux de données *Last.fm* n'ont pas d'évaluations explicites des utilisateurs. On y retrouve le nombre de fois qu'un utilisateur a écouté un artiste ou un morceau. Ces jeux de données, en raison de leur nature, ont influencé les modèles de recommandations qui ont été créés à partir d'eux. Dans la section 2.1, nous discuterons plus en détail de cette problématique et nous démontrerons que l'industrie a aussi influencé la recherche dans certains aspects de la recommandation. Dans la section 2.2, nous présenterons les limites des systèmes de recommandation, notamment ceux basés exclusivement sur la précision.

1. http://grouplens.org/datasets/
2. http://ocelma.net/MusicRecommendationDataset/
3. http://labrosa.ee.columbia.edu/millionsong/

## 2.1 De la prédiction des évaluations au Top-N

La prédiction des évaluations est un problème qui a beaucoup été abordé dans les systèmes de recommandation [Gunawardana et Shani, 2015]. Ce problème peut être posé de la manière suivante : les interactions entre les utilisateurs et les articles sont stockées dans une matrice $R$ d'évaluation et sont souvent encodées sous la forme d'entiers allant de 1 à 5 étoiles avec 1 la valeur minimale signifiant que l'utilisateur n'a pas apprécié l'article et 5 signifiant que l'utilisateur a particulièrement apprécié l'article. Le but du système de recommandation $S$ est de prédire les futures évaluations des utilisateurs. Le processus est le suivant : le jeu de données est divisé en deux, un ensemble d'entrainement (*Train*) sur lequel l'algorithme de recommandation apprend à prédire la note, et un second ensemble (*Test*) sur lequel l'algorithme teste ces prédictions. Les métriques utilisées pour valider la précision de la prédiction sont les suivantes : l'erreur moyenne absolue (*MAE*, formule 2.1) et l'erreur quadratique moyenne (*RMSE*, formule 2.2).

$$\mathrm{MAE}(S) = \frac{1}{|R_{test}|} \sum_{(u,i)\in R_{test}} |r_{u,i} - s(u,i)| \tag{2.1}$$

$$\mathrm{RMSE}(S) = \sqrt{\frac{1}{|R_{test}|} \sum_{(u,i)\in R_{test}} (r_{u,i} - s(u,i))^2} \tag{2.2}$$

L'optimisation de ce système est de minimiser l'erreur de prédiction, c'est-à-dire arriver à des valeurs de *MAE* et *RMSE* minimales [Sarwar *et al.*, 2001; Wei *et al.*, 2012]. Les recherches sur les prédictions sont corrélées à la nature des jeux de données disponibles. Les jeux de *MovieLens* sont composés des appréciations explicites que les utilisateurs ont données aux films. Initialement, la majorité des systèmes de recommandation sont des systèmes prédictifs, c'est-à-dire qu'ils prédisent une opinion ou la probabilité d'un achat. L'hypothèse est que les recommandations produites par un système ayant ces valeurs MAE ou RMSE minimales, sera le système qui apportera les meilleures recommandations aux utilisateurs.

Le *Netflix Prize* a amplifié cet engouement pour la prédiction des appréciations [Parra et Sahebi, 2013]. Dans ce qui est considéré comme une grande contribution à la communauté des chercheurs, la compagnie *Netflix Inc.* a rendu public un jeu de données (qui n'est plus disponible), et offert un prix d'1 million de dollars *US* à la personne ou à l'équipe qui arriverait à modéliser le jeu de données et à produire un algorithme de prédiction meilleur que le leur. Il fallait améliorer le score de leur système appelé *Cinematch* d'au moins 10%. Plus formellement, il fallait proposer le modèle qui améliorait le *RMSE* de *Cinematch* qui était de 0.9525, avec comme performance attendue 0.8572. Le jeu de données était composé de 100 millions d'évaluations datées effectuées sur 18 000 films, provenant de 480 000 utilisateurs anonymes choisis aléatoirement. Les données ont été collectées entre octobre 1998 et décembre 2005. Le jeu de données était aussi composé d'un ensemble test de 3 millions d'évaluations, choisies parmi les plus récentes [Bennett et Lanning, 2007]. Cette compétition a permis de faire des avancées conséquentes sur le filtrage collaboratif [Koren, 2008; Koren *et al.*, 2009; Takács *et al.*, 2008; Zhou *et al.*, 2008; Koren,

2010], notamment en employant la factorisation de matrices et les machines de Boltzmann restreintes. On peut avancer que les jeux de données disponibles, plus la notoriété du *Netflix Prize* a participé à créer le mythe de la précision.

Cependant des publications scientifiques ont aussi été produites pour dire que les méthodes de prédiction d'évaluation n'étaient pas appropriées à tous les cas de recommandation. Améliorer la prédiction n'améliorait pas systématiquement ni l'utilité des recommandations, ni l'expérience utilisateur [McNee *et al.*, 2006]. Sur *le blog de Netflix* [4], on peut lire que la recherche de modèles de prédiction efficace à travers le *Netflix Prize* répondait à un besoin spécifique. En effet en 2006 *Netflix* était un système de location de *DVD* par service postal disponible aux États-Unis. Leur but était d'aider les gens à remplir leur liste de demandes de films, qu'ils allaient par la suite recevoir dans quelques jours pour les visionner. Les clients devaient sélectionner les films avec attention, car l'échange d'un *DVD* par un autre pouvait durer plusieurs jours. Depuis, *Netflix* est passé au *streaming* (en 2007, 1 an après avoir lancé le prix) et la problématique n'est plus la même : les utilisateurs recherchent quelque chose à regarder et peuvent regarder des extraits avant d'en choisir un. De plus, les retours utilisateurs sont faits en direct, et les administrateurs du système peuvent vérifier qu'un utilisateur a regardé un film en entier ou non. Pour ces raisons et peut-être d'autres, *Netflix* n'a pas mis en production l'algorithme gagnant, mais a quand même couronné l'équipe gagnante *Netflix Leaderboard* [5].

Outre la prédiction, un deuxième pan de la recherche s'intéresse à choisir une sous-partie d'un catalogue et à la proposer aux utilisateurs. Ce problème se présente généralement sous l'appellation *Top-n* ou *Top-k* [Cremonesi *et al.*, 2010], $n$ ou $k$ est souvent un nombre « petit ». Formellement, la recommandation est étendue à la tâche suivante : pour chaque utilisateur, retrouver la liste d'articles $S_u$ de longueur $k$ qui pourrait l'intéresser [Breese *et al.*, 1998; Khabbaz et Lakshmanan, 2011; Liu et Aberer, 2014]. Le classement de la liste est effectué en fonction du score $s(u, i)$ obtenu par les articles. Plusieurs stratégies ont été proposées pour sélectionner les listes de recommandations. Une solution triviale serait pour les approches du filtrage collaboratif de convertir les prédictions d'évaluations en liste *Top-n* [Cremonesi *et al.*, 2010] et pour les approches basées sur le contenu de ranger la liste *Top-n* selon le score obtenu par la mesure de similarité.

Une autre manière de résoudre ce problème est de le formuler comme un problème d'apprentissage. Il suffit de sélectionner des exemples positifs et négatifs des données historiques des utilisateurs et d'utiliser un algorithme d'apprentissage automatique pour apprendre les poids qui permettraient d'optimiser un objectif (bien ranger les articles). Ce problème est connu sous le nom de *Learning to rank* et a aussi été abordé dans les systèmes de recherche d'informations [Liu, 2009]. La notion de personnalisation est centrale aux systèmes de recommandation dans les algorithmes *Learning to rank*. Ces systèmes cherchent à optimiser des modèles pour chaque utilisateur [Karatzoglou *et al.*, 2013].

4. `https://goo.gl/zVJrJe`
5. `http://www.netflixprize.com/leaderboard.html`

Le but du système de recommandation maintenant est de présenter un nombre $n$ d'articles à un utilisateur. La manière commune de présenter les recommandations de nos jours sur les plateformes est sous la forme de liste. La qualité de la liste de recommandations produite peut être évaluée en utilisant des métriques empruntées à la recherche d'informations, telles que la précision, le rappel [Bellogín *et al.*, 2011], et la mesure de pertinence graduelle *Normalized Discounted Cumulative Gain (nDCG)* [Järvelin et Kekäläinen, 2002]. L'ensemble des interactions entre utilisateurs et articles est divisé pour chaque utilisateur en deux ensembles, *train* et *test*. L'algorithme produit des listes de recommandations et le contenu de ces listes est comparé au contenu des listes de *test* en utilisant les métriques citées préalablement. Cette manière de procéder permet aussi de mesurer d'autres aspects de la recommandation, tels que la nouveauté ou la diversité.

## 2.2 Les limites de la précision

Traditionnellement, les recherches sur les systèmes de recommandation ont été orientées vers la maximisation de la précision (l'exactitude) des recommandations, c'est-à-dire retrouver les articles les plus pertinents possibles. Mais certains articles de recherche ont montré qu'il existe d'autres facteurs qui améliorent l'utilité des recommandations et la satisfaction des utilisateurs. La précision seule n'offrirait pas aux utilisateurs des systèmes de recommandation une expérience satisfaisante [Herlocker *et al.*, 2004]. La recherche effrénée de l'amélioration de la précision aurait même nui aux systèmes de recommandation [McNee *et al.*, 2006]. La satisfaction des utilisateurs et la performance du système dépendent d'autres propriétés, comme la prise en compte de la nouveauté, de la diversité, de l'aptitude du système à créer de la sérendipité et sa capacité à couvrir l'entièreté du catalogue [McNee *et al.*, 2006; Ge *et al.*, 2010].

En règle générale, les systèmes de recommandation basés sur le contenu proposent des contenus jugés trop semblables à ceux que l'utilisateur a consommés dans le passé [Adamopoulos et Tuzhilin, 2014]. La similarité reste un élément important pour la caractérisation des articles qui peuvent intéresser un utilisateur. Cependant, exposer des articles trop semblables apporte peu de valeurs aux recommandations faites. En effet, ces recommandations apportent peu de nouveauté, et limitent la découverte [Ziegler *et al.*, 2005]. Par ailleurs, les systèmes de recommandation basés sur le filtrage collaboratif ont tendance à proposer des articles populaires [Steck, 2011]. Ce phénomène est appelé *The Harry Potter effect* [6], et touche particulièrement les modèles collaboratifs basés sur l'article (section 1.2.1) [Adomavicius et Kwon, 2009]. Un système de recommandation de livre prendra peu de risque en recommandant *Harry Potter* à tous les utilisateurs, capitalisant sur le fait que beaucoup de personnes auront apprécié le livre. Les plus populaires deviennent plus populaires, et ce biais de concentration peut annihiler des propositions de certains articles qui seraient plus pertinents et plus utiles pour des consommateurs [Fleder et Hosanagar, 2009a]. Cela aura comme effet de creuser encore plus l'écart entre les éléments en

6. `http://recsyswiki.com/wiki/Harry_Potter_effect`


FIGURE 2.1 – Le principe de la longue traîne [Anderson, 2008].

tête de la courbe de la longue traîne et ceux moins populaires qui sont dans la queue de la courbe de longue traîne.

La longue traîne (*Long tail*) est un concept défini en 2004 par le rédacteur en chef de la revue américaine *Wired*, Chris Anderson. Il écrit par la suite un livre sur le concept intitulé *The Long Tail : Why the Future of Business Is Selling Less of More* [Anderson, 2008]. Il constate en rapportant les statistiques de ventes des sites commerciaux comme *Amazon* que les *bestsellers* représentent peu d'ouvrages, mais beaucoup de ventes, et que le reste des références est composé de beaucoup d'ouvrages, très peu vendus individuellement, mais dont la masse cumulée représente l'essentiel des bénéfices du site de commerce en ligne. Si nous représentons la distribution des ventes dans un espace à deux dimensions, l'axe X représente les articles, l'axe Y représente les ventes, on remarque que la tête de la courbe est composée de peu d'articles (elle représente les *hits*, les *Blockbuster*, tels que Star Wars, The Beatles...) puis elle commence à diminuer jusqu'à tendre vers 0 (on y retrouve les nouveaux articles, les articles peu populaires, les artistes indépendants, le groupe techno de vos amis...), figure 2.1. Vous pouvez voir cette courbe comme la silhouette d'un brontosaure, avec une queue très longue. Dans son livre Chris Anderson défendait l'économie de la longue traîne, et la présentait même comme le futur des ventes en ligne.

Des simulations ont été faites sur l'effet qu'ont les systèmes de recommandation sur la diversification des ventes et donc la capacité à puiser des articles dans la longue traîne [Fleder et Hosanagar, 2009b]. Les auteurs ont soutenu que tous les utilisateurs sont dirigés vers une expérience commune. Les systèmes de recommandation n'arrivent pas à recommander des

articles peu évalués (c.-à-d. ayant peu de notes d'appréciations) même si ces articles pourraient s'avérer utiles aux utilisateurs. En revanche, les articles recommandés à un utilisateur peuvent être nouveaux pour lui (c.-à-d. des articles qu'il n'a jamais vus ou différents de ses goûts), mais quand on étudie les recommandations dans leur ensemble on remarque qu'elles sont les mêmes pour tout le monde (c.-à-d. les plus populaires). De plus, une étude faite sur deux groupes d'utilisateurs d'*iTunes* a démontré que les utilisateurs consomment les mêmes articles [Hosanagar *et al.*, 2014]. Les auteurs ont affirmé que les systèmes de recommandation actuels de type filtrage collaboratif, en aidant les utilisateurs à étendre leurs intérêts, augmentaient la probabilité qu'ils consomment tous des articles semblables.

Si les algorithmes ont tendance à proposer exclusivement des articles similaires, ils peuvent aussi avoir un effet pervers et enfermer les utilisateurs dans des bulles de filtre [Pariser, 2011]. La bulle de filtre (*Filter Bubble*) est un concept présenté par Eli Pariser : les informations qui sont filtrées pour un utilisateur sont conformes à ses idées préconçues du monde. L'information qui lui est présentée lui est agréable, et l'information qui devrait l'inciter à réfléchir ou à remettre en cause ses hypothèses est éloignée de lui. Dans son livre Elie Pariser critique la personnalisation du *web* orchestrée par les systèmes de recherche et de recommandation. Les recherches sur *Google* sont personnalisées afin de mieux répondre à nos attentes, *Amazon* voudrait nous recommander des livres avant même que nous les ayons commandés. Des résultats empiriques ont prouvé que les systèmes de recommandation présentaient assez souvent ces symptômes [Nguyen *et al.*, 2014]. Pour reprendre Dominique Cardon [Cardon, 2015],

> ces manières de présenter l'information enferment les individus dans la bulle de leur choix, plient leur destin dans l'entonnoir du probable et nourrissent la précision du ciblage d'une capture disproportionnée d'informations personnelles

. Néanmoins, certains utilisateurs du *web* créent leur bulle de filtre en sélectionnant eux-mêmes leurs amis sur les réseaux sociaux ainsi que les articles qu'ils lisent [Bakshy *et al.*]. Les algorithmes de filtrage collaboratif ne feraient qu'être rationnels en ne se basant que sur les actions posées par les individus sur le *web*. Cardon [2015] avance que la prédiction des algorithmes ne fait qu'exploiter la régularité des structures de goûts des utilisateurs. Il existe un caractère régulier et prévisible des pratiques de lecture, d'écoute, etc. Les algorithmes prédictifs se basent sur le passé. Si un utilisateur a des goûts variés, il aura des recommandations variées, s'il a des goûts *mainstream*, il aura des recommandations *mainstream*. Les algorithmes font constamment l'hypothèse que le futur de l'internaute sera une reproduction de son passé.

Il est nécessaire de souligner ces limites des systèmes de recommandation. Selon Pariser, un monde qui est construit autour de ce qui est familier est un monde où il n'y a rien à apprendre. Comme Pariser l'a rapporté dans son livre « The Filter Bubble : What the Internet Is Hiding from You », l'écrivain et professeur américain Yochai Benkler soutient qu'un individu doit être tenu informé de l'ensemble des différentes options qui sont à sa disposition, ainsi que l'ensemble des différents choix de vie. Ainsi, dans le monde des systèmes de recommandation, il faut proposer des systèmes alternatifs, qui viseraient à contrer l'effet de masse. Il s'agit de trouver des

techniques de recommandations qui aboutissent à des propositions de consommations diverses, nouvelles, et qui maximisent la couverture du catalogue, c'est-à-dire de l'ensemble du possible.

Une bonne recommandation ne dépend donc pas seulement de l'aptitude que le système a à prédire les évaluations passées. Même si la perception de la qualité d'une liste de recommandation est subjective, que les goûts des utilisateurs changent, et que ces utilisateurs peuvent être influencés par des facteurs externes aux systèmes, il reste nécessaire d'explorer d'autres dimensions pour continuer à proposer de nouvelles mesures pour qualifier les recommandations. Dans le chapitre suivant (chapitre 3) nous allons présenter les différentes techniques mises en place pour aller au-delà du calcul de la précision, et répondre aux problématiques de surconcentration et de personnalisation accrue.

Chapitre 3

# Au-delà de la précision

L'utilité des recommandations pour un utilisateur va au-delà de la similarité et de la prédiction précise des appréciations. Dès 2006 la diversité et la nouveauté dans les listes de recommandations ont commencé à être vues comme des facteurs qui peuvent apporter une valeur ajoutée aux recommandations [McNee *et al.*, 2006]. L'utilisateur ressent parfois le désir de découvrir des expériences qui lui sont jusque là inconnues. Il veut briser sa routine. Cette expérience recherchée peut être en soi agréable [McAlister et Pessemier, 1982]. De plus cette recherche de nouveauté et de diversité permettrait de contrer la bulle de filtre de Pariser [2011], résultat d'une personnalisation accrue du *Web* comme nous l'avons décrit au chapitre 2. Des travaux sur les systèmes de recommandation ont montré que les utilisateurs étaient attirés par des listes d'articles diversifiées [Ferwerda *et al.*, 2017]. Des listes de recommandations qui répondent à cette caractéristique ont aussi une utilité pour les systèmes de recommandation : la diversité peut être considérée pour contrer l'incertitude quant au besoin réel de l'utilisateur. Du point de vue du système, les actions de l'utilisateur sont considérées comme une preuve implicite de ses besoins. En revanche, l'identification des préférences réelles de l'utilisateur reste imprécise. La généralisation des préférences d'un utilisateur à partir du peu d'éléments avec lesquels il a interagi implique une ambigüité et une incertitude considérable. De plus les préférences d'un utilisateur varient avec le temps, dépendent d'un contexte et sont souvent contradictoires. Ainsi, la diversité peut être considérée pour contrer cette incertitude. En proposant une liste de recommandations comportant des articles diversifiés, le système de recommandation augmente la probabilité que l'utilisateur apprécie au moins un article. Le système prend moins de risques en adoptant cette stratégie plutôt qu'en cherchant l'article qui répond à toutes les facettes des préférences de l'utilisateur.

Le modèle d'affaire de l'entreprise détentrice du système de recommandation peut aussi influencer le choix de la proportion de nouveauté et de diversité à incorporer dans les listes de recommandations. Une entreprise peut choisir de tabler sur la vente des articles de niches se trouvant dans la longue traîne de son catalogue. Les bénéfices à tirer de la vente des articles de niches peuvent être comparables aux bénéfices de la vente des quelques articles *stars* du catalogue. Pour ce faire, son système de recommandation doit contrer les effets de masse et de personnalisation accrue.

Les considérations citées ci-dessus peuvent ne pas correspondre à tous les domaines de consommation. Parfois, la diversité et la nouveauté ne sont pas des facteurs importants à prendre en compte lors de la création de listes de recommandations. Dans le domaine de la santé, la

précision reste un facteur important, par exemple pour la recommandation de traitement pour les patients atteints de maladie de la peau [Gräßer *et al.*, 2017]. Dans le reste de ce chapitre, nous allons définir et présenter les concepts qui vont au-delà de la précision : la nouveauté et la diversité.

## 3.1 Les méthodes d'introduction de la nouveauté

La nouveauté peut être comprise comme la différence entre les expériences passées et les expériences actuelles. Un article nouveau pour un utilisateur est un article qui lui est inconnu. Toutefois, la notion de nouveauté peut prendre des formes particulières selon l'approche. Par exemple, un utilisateur d'un site de streaming de musique à qui le système recommande un morceau qu'il n'a jamais écouté auparavant peut interpréter cette proposition comme une recommandation nouvelle. En revanche si le morceau proposé provient d'un artiste populaire, la nouveauté est moindre que dans le cas d'un artiste ou d'un genre musical inconnu puisé dans la longue traîne [Castells *et al.*, 2015]. Il existe donc différentes façons d'aborder le concept de nouveauté. Une définition simple peut consister à vérifier la présence de l'article dans l'historique de l'utilisateur. La nouveauté prendra alors une valeur binaire [Vargas et Castells, 2011]. Il y a aussi la nouveauté par rapport aux articles de la longue traîne [Park et Tuzhilin, 2008; Celma et Herrera, 2008]. Ici, on va chercher en considérant le nombre d'utilisateurs qui ont interagi avec l'article, à créer de la surprise, à proposer une expérience inattendue, en recommandant les articles peu connus. Une notion proche de cette définition de la nouveauté est la sérendipité. Le système cherche à provoquer pour l'utilisateur une découverte heureuse. Une réaction émotionnelle positive de l'utilisateur est souhaitée dans ce cas. La sérendipité formellement est une recommandation d'un article nouveau, inconnu, pertinent et qui provoque une réaction émotionnelle positive de l'utilisateur [Murakami *et al.*, 2007; Zhang *et al.*, 2012].

### 3.1.1 La nouveauté dans les systèmes de recommandation

Nous allons utiliser la notation usuelle (tableau 3.1) pour présenter les méthodes et formules de cette partie, ce tableau étend les tableaux 1.1, 1.2.

| Paramètres | Explication |
|---|---|
| $I$ | ensemble des articles |
| $U$ | ensemble des utilisateurs |
| $k$ | longueur de la liste de recommandations |
| $S_u$ | liste de recommandation pour un utilisateur $u$ |
| $Dist(i, j)$ | distance entre un item $i$ et un item $j$ |

Tableau 3.1 – Notation et paramètres utilisés.

#### 3.1.1.1 La nouveauté par rapport à la longue traîne

La recommandation par rapport à la longue traîne se base sur la popularité de l'article pour estimer sa nouveauté [Park et Tuzhilin, 2008; Celma et Herrera, 2008; Zhou *et al.*, 2010]. La nouveauté est vue comme l'inverse de la popularité et est modélisée comme la probabilité qu'un utilisateur pris au hasard connaisse un article [Zhou *et al.*, 2010]. Cette manière d'apporter de la nouveauté fait remonter des articles de la longue traîne et permet de lutter contre l'effet Harry Potter (section 2.2). Il y a plus de chances de retrouver un élément nouveau pour l'utilisateur dans la longue traîne que parmi les éléments les plus populaires. En effet, l'utilisateur entendra parler des éléments populaires sans doute par d'autres moyens. Pour pénaliser la popularité, Zhou *et al.* [2010] utilisent le négatif d'une fonction logarithmique. Cette fonction est analogue à la fonction $IDF$ (l'inverse de la fréquence des documents) des modèles de recherche d'information avec les articles à la place des termes et les utilisateurs à la place des documents ($IUF = -log_2 \frac{|U_i|}{U}$). La nouveauté dans une liste d'un système de recommandation peut être calculée comme la moyenne des $IUF$ (formule 3.1). L'optimisation de la nouveauté par rapport à la longue traîne est résolue en utilisant une stratégie hybride qui combine des méthodes de filtrage collaboratif aux techniques de diffusion de graphes.

$$\text{MIUF} = -\frac{1}{|S| \sum_{i \in S} log_2 \frac{|U_i|}{U}} \tag{3.1}$$

Pour accéder aux éléments de la longue traîne et proposer des artistes nouveaux, d'autres ont étudié la topologie des réseaux formés par un système de recommandation de filtrage collaboratif basé sur l'article et un système de recommandation basé sur le contenu [Celma et Herrera, 2008]. Dans ces travaux effectués sur un jeu de données musicales, il a été montré que la topologie du réseau basé sur le filtrage collaboratif ne permettait pas de produire des recommandations nouvelles. Ces systèmes de recommandation ne permettaient pas de faire des découvertes, tandis que les systèmes basés sur le contenu permettaient de proposer des découvertes à l'utilisateur grâce à leur topologie. Mais une étude basée sur les utilisateurs a montré que les utilisateurs préfèrent les recommandations du système basé sur l'article. La solution proposée serait de passer d'un système à un autre selon les envies de l'utilisateur. Un utilisateur qui est dans une phase de découverte verra des recommandations du modèle basé sur le contenu, tandis qu'un utilisateur qui veut écouter ses morceaux favoris verra des recommandations du modèle basé sur le filtrage collaboratif. La difficulté réside dans le profilage de l'utilisateur et la détection de ces modes. Ces expériences ont prouvé que la popularité filtrait les articles de mauvaise qualité en les rejetant dans la longue traîne, bien que la popularité ne soit pas gage de qualité [Salganik *et al.*, 2006]. L'identification des articles de qualité peut se faire par l'identification d'experts dans le réseau. Le système se basera sur leur goût pour produire des recommandations apportant de la nouveauté. Ce concept a été employé et a permis d'identifier ces experts ainsi que des novices, et après à un partitionnement (*clustering*) des articles par leur similarité, d'assigner les utilisateurs experts

aux *clusters* dans lesquels ils ont le plus interagi avec les articles. Une recommandation pour un utilisateur se fera en identifiant les *clusters* pour lesquels il est novice, puis en se basant sur les connaissances des experts du *cluster* pour identifier des articles intéressants [Lee et Lee, 2013]. Pour proposer des articles nouveaux aux utilisateurs, Menk *et al.* [2017] ont modélisé des aspects de la personnalité des utilisateurs en fonction de leur comportement passé, de leurs études, et d'autres aspects démographiques. Ces méthodes se rapprochent du concept *Unexpectedness*, que nous traduisons par l'inattendu en français.

#### 3.1.1.2 Une autre forme de nouveauté : l'inattendu

L'inattendu (*Unexpectedness*) ([Adamopoulos et Tuzhilin, 2014]) est lié au fait qu'un utilisateur reçoive des articles différents de ceux qui se trouvent dans son historique, des articles dont il n'est pas familier, des articles éloignés de ce qu'il attend. Cette variante de la nouveauté se rapproche de la sérendipité, et diffère de la nouveauté par rapport à la longue traîne, car elle considère les expériences passées de l'utilisateur pour mesurer l'impact qu'un article aura sur lui en ce qui concerne la nouveauté. La nouveauté par rapport à la longue traîne est indépendante de l'utilisateur considéré. Or en règle général, le degré de nouveauté d'un article peut varier d'un utilisateur à l'autre. Ce n'est pas parce qu'un article se trouve dans la longue traîne qu'il est forcément nouveau pour tous les utilisateurs. Certains utilisateurs éclairés n'ont pas trouvé les recommandations de la longue traîne de *Last.fm* « nouvelles » [Celma et Herrera, 2008]. Pour déterminer la nouveauté d'un article, on peut considérer l'article ou les caractéristiques de l'article et vérifier que l'article n'a pas déjà été consommé par l'utilisateur, ou que l'utilisateur n'a pas consommé d'articles ayant des caractéristiques semblables [Billsus et Pazzani, 2000]. Cette forme de nouveauté par rapport aux attributs de l'article est relativement simple à mesurer. La nouveauté d'un article par rapport aux articles de l'historique de l'utilisateur peut être mesurée à partir d'une fonction de distance [Castells *et al.*, 2015]. Cette fonction permettra de connaitre le degré de surprise qu'un article peut apporter à un utilisateur. Pour une liste de recommandations, cette valeur peut être mesurée en prenant la moyenne de la somme des distances [Castells *et al.*, 2015] (Formule 3.2).

$$\text{Unexp} = \frac{1}{|S||I_u|} \sum_{(i)\in S} \sum_{(j)\in I_u} Dist(i, j) \tag{3.2}$$

L'inattendu est aussi défini comme l'écart par rapport aux résultats obtenus à partir d'un modèle de prédiction « primitif » et ceux d'un autre algorithme qui favorise la sérendipité et l'inattendu [Murakami *et al.*, 2007; Ge *et al.*, 2010]. Un modèle « primitif » est un modèle qui produit des recommandations peu surprenantes, p.ex. un système qui propose que les articles populaires appréciés par l'utilisateur, ou les articles définis par les genres préférés de l'utilisateur. L'inattendu produit par une liste de recommandation est donné par la formule 3.3 [Ge *et al.*,

2010], avec $EX$ (*expected*) l'ensemble des articles recommandés par la méthode primitive.

$$\text{Unexp}(S) = \frac{|S \setminus EX|}{|S|} \tag{3.3}$$

Ces méthodes diffèrent de la définition de l'inattendu selon Adamopoulos et Tuzhilin [2014]. Pour ces derniers, pour produire quelque chose d'inattendu, il faut d'abord connaitre les attentes de l'utilisateur ($E$). Les attentes d'un utilisateur sont les articles qu'il a déjà consommés, ou les recommandations typiques qu'il pourrait avoir. Ces recommandations typiques sont les articles semblables ou liés aux articles se trouvant dans son historique de consommation. Ils ont ainsi défini l'utilité (*Util*) de la recommandation comme une fonction de la qualité de la recommandation. Leur formule est différente de la formule 3.3 parce qu'elle utilise l'ensemble des articles attendus et non les articles recommandés par la méthode primitive (formule 3.4). Les auteurs ont aussi proposé une variante dans laquelle la distance des articles dans la liste de recommandations par rapport aux articles attendus est prise comme mesure de l'inattendu.

$$\text{Unexp}(S) = \frac{|(S_u \setminus E_u) \cap Util_u|}{|S|} \tag{3.4}$$

### 3.1.2 La sérendipité

La sérendipité, selon le dictionnaire *Larousse* [1] est la capacité, l'art de faire une découverte scientifique notamment par hasard. Cette définition reprend la définition d'Horace Walpole de 1754. Parmi les nombreux exemples de découvertes liées au hasard on peut citer : la pénicilline (par Fleming), le Post-it, l'Amérique (par Christophe Colomb)... L'origine du mot « sérendipité » provient du conte merveilleux Persan « *Les trois princes de Sérendip* » [Andel, 1994]. Selon Royston M. Roberts, la sérendipité dépend de la personne qui recherche l'information, de son attitude, de son ouverture d'esprit, de sa curiosité et de sa culture. Certains disent qu'il est impossible de faire un programme produisant de la sérendipité en raison de sa nature propre, c'est-à-dire du hasard dont les conséquences sont heureuses [Andel, 1994], mais d'autres disent qu'il est possible de réaliser des programmes permettant de favoriser l'apparition de la sérendipité [Iaquinta *et al.*, 2008].

La sérendipité dans les systèmes de recommandation est souvent définie comme une recommandation qui produit une réaction émotionnelle positive de l'utilisateur [Murakami *et al.*, 2007; Zhang *et al.*, 2012]. C'est un concept qui est proche du concept de nouveauté, mais qui s'en écarte par son côté « hasardeux ». La recommandation est jugée fortuite (*serendipitous recommandation*) si l'article proposé est inconnu de l'utilisateur, s'il lui était difficile, voire impossible de le retrouver seul [Herlocker *et al.*, 2004]. Cette dernière définition a été adoptée par Iaquinta *et al.* [2008] pour tenter de provoquer la sérendipité dans un système de recommandation basé sur le contenu. Les auteurs se sont servis de l'incertitude de certaines prédictions d'un classifieur binaire (aime, n'aime pas) pour induire de la sérendipité. Il était difficile pour leur

1. `http://www.larousse.fr/dictionnaires/francais/sérendipité/186748`

classifieur d'apprendre la classe de certains contenus. Ils ont utilisé ces contenus pour proposer des recommandations fortuites. Ils se sont rendu compte au cours d'expériences humaines qu'en choisissant des contenus au hasard dans la liste de contenu dont la classe est incertaine, ils amélioraient la satisfaction des utilisateurs. En revanche, les auteurs n'étaient pas sûrs que les thèmes abordés dans les documents recommandés étaient vraiment inconnus des utilisateurs.

Les métriques de distance pourraient aussi permettre d'atteindre les zones qui seraient propices à la sérendipité dans un jeu de données. À partir de données récoltées sur des utilisateurs, il a été possible de voir que la zone des articles ou les recommandations fortuites apparaissaient, était plus éloignée et différente de la zone des éléments non appréciés par les utilisateurs [Akiyama *et al.*, 2010]. Les auteurs de cette étude ont justement produit une métrique de distance leur permettant d'atteindre cette zone. Similairement à cette approche et dans le domaine musical, Zhang *et al.* [2012] ont présenté un système de recommandation dont l'un des objectifs était de définir des *clusters* d'articles à partir des préférences des utilisateurs et de proposer des recommandations qui se trouvent aux frontières de ces *clusters*. Leur approche améliorait la nouveauté, la sérendipité et la diversité sans trop pénaliser la précision, et leurs résultats démontraient aussi l'existence d'une zone pour la sérendipité. Certains travaux n'ont pas cherché à identifier une zone qui permet de produire des recommandations fortuites, mais on choisit de produire des recommandations fortuites pour un utilisateur en appliquant des méthodes de marche aléatoire dans un graphe de taxonomie à partir d'utilisateurs semblables [Nakatsuji *et al.*, 2010]. Cette méthode requiert d'organiser les articles dans une taxonomie. Taramigkou *et al.* [2013] ont demandé explicitement aux utilisateurs d'une plateforme musicale de sélectionner les genres musicaux qui les intéressent afin de trouver des utilisateurs experts de ces genres pour produire des recommandations. Toujours dans le domaine musical, Ziegler *et al.* [2014] ont aussi proposé d'ajouter de la sérendipité en se basant sur les écoutes des autres utilisateurs. Les auteurs ont choisi d'utiliser l'historique des 5 derniers utilisateurs ayant écouté les 5 derniers morceaux de l'utilisateur afin d'ajouter à une liste de recommandation standard de la sérendipité. Gemmis *et al.* [2015b] se sont basés sur des connaissances externes pour enrichir un modèle de recommandation basé sur l'analyse de graphes. Ils ont montré qu'en incorporant des connaissances provenant de *Wikipedia*, ils arrivaient à produire des recommandations fortuites sans impacter négativement la précision. La sérendipité amenée par cette méthode ne semble pas être personnalisée. Un travail plus récent s'est basé sur une théorie psychologique pour mesurer la curiosité des utilisateurs [Maccatrozzo *et al.*, 2017]. Les auteurs sont ainsi capables d'estimer la proportion des utilisateurs prêts à recevoir des articles externes à leur zone de confort, et aussi la proportion d'articles allant vers de la sérendipité qu'ils peuvent mettre dans les recommandations.

Somme toute, la nouveauté dans les recommandations est définie comme la prescription d'articles qui sortent des habitudes des utilisateurs, ou d'articles peu populaires. Les systèmes doivent prendre en compte le profil de l'utilisateur et son envie de découvrir. Ces méthodes ont été développées dans le but de sortir les utilisateurs de leur bulle de filtre. D'autres méthodes ont été abordées pour répondre à la personnalisation accrue, telle que la diversité (section 3.2).

## 3.2 Les méthodes de diversification

La diversification a été introduite comme une des solutions possibles pour répondre au problème de la personnalisation accrue et aussi pour permettre à l'utilisateur d'étendre son profil. La diversité a été définie comme étant l'opposée de la similarité [Bradley et Smyth, 2001]. La diversité est définie de manière semblable comme la différence interne qu'il existe entre les éléments faisant partie de l'expérience proposée à un utilisateur [Castells *et al.*, 2015]. Une liste de recommandations sera considérée comme diverse si elle inclut des articles de genres différents plutôt que plusieurs articles du même genre, indépendamment du fait que les articles soient originaux pour l'utilisateur. La diversité et la nouveauté sont deux concepts liés, ils cherchent tous les deux à produire une différence dans la liste de recommandations. La différence dans le cas de la nouveauté se situe entre les expériences passées et l'expérience proposée, et dans le cas de la diversité la différence se situe au sein même de l'expérience proposée.

La diversité a été beaucoup abordée dans la recherche d'information durant la dernière décennie et a permis de dégager une base théorique, des métriques et méthodes d'évaluation [Carbonell et Goldstein, 1998; Chen et Karger, 2006; Agrawal *et al.*, 2009; Chapelle *et al.*, 2011]. Les techniques développées dans ce champ de recherche ont été utilisées dans les systèmes de recommandation [Vargas *et al.*, 2012]. D'autres champs de recherche comme l'écologie ont étudié la diversité et ont formalisé le problème tout en définissant et comparant un large éventail de métriques de diversité, tels que l'indice de Gini-Simpson et l'entropie [Patil et Taillie, 1982]. Ces métriques sont aussi utilisées dans les méthodes de diversification des listes de recommandations. En économie, la diversité a aussi été étudiée (diversité contre oligopole). Elle porte sur la diversité des secteurs dans lesquels une entreprise a un marché, la variété des articles, la diversité des investissements [Lubatkin et Chatterjee, 1994]. Les concepts dégagés sont fortement liés aux stratégies des systèmes de recommandation.

### 3.2.1 Les différents types de diversité

Dans la littérature des systèmes de recommandation, on retrouve deux types de diversité : l'une basée sur la moyenne des distances des articles d'une liste de recommandations et l'autre sur la diversité moyenne entre les listes de recommandation produite par le système. La moyenne des distances des articles d'une liste de recommandations *Intra-List Diversity* est la première métrique proposée. Elle est donnée par la formule 3.5 [Smyth et McClave, 2001; Ziegler *et al.*, 2005; Zhang et Hurley, 2008].

La distance entre les articles est primordiale dans la formule 3.5, et aussi paramétrable. Il existe beaucoup de métriques de distance dans la littérature. Les attributs des objets sont souvent utilisés pour estimer la distance [Ziegler *et al.*, 2005] mais les interactions des utilisateurs peuvent aussi être utilisées pour estimer cette distance [Ribeiro *et al.*, 2014]. La diversité moyenne entre les listes de recommandation a aussi été proposée pour mesurer la quantité d'articles recommandés aux utilisateurs [Adomavicius et Kwon, 2012, 2014] (formule 3.6). Cette

métrique est différente de la première citée, et elle est aussi connue sous le nom d'*item coverage*, que nous traduisons par le nombre d'articles couverts [Herlocker *et al.*, 1999, 2004; Bellogín *et al.*, 2013].

$$\text{ILD} = \frac{1}{|S|(|S|-1)} \sum_{i \in S} \sum_{j \in S} Dist(i, j) \tag{3.5}$$

$$\text{AggDiv} = |\bigcup_{u \in U} S_u| \tag{3.6}$$

La diversité moyenne entre les listes de recommandation a aussi été mesurée en utilisant le coefficient de Gini, l'index de Gini-Simpsons, l'entropie Patil et Taillie [1982] qui permettent d'obtenir des mesures statistiques sur par exemple la dispersion en biologie (biodiversité) ou la distribution des inégalités en économie. Ces mesures permettent de voir aussi comment sont réparties les recommandations. L'index de Gini a été formulé par Shani et Gunawardana [2011] (formule 3.7).

$$\text{Gini}(RecSys) = \frac{1}{|I|-1} \sum_{k=1}^{|I|} (2k - |I| - 1) p(i_k | RecSys) \tag{3.7}$$

avec $p(i_k|RecSys)$ la probabilité que l'item $k$ soit recommandé par le système de recommandation *RecSys* :

$$\text{Gini}(RecSys) = \frac{|u \in U : i \in S_u|}{\sum_{u \in U} |S_u|} \tag{3.8}$$

Les approches présentées ici permettent aussi de diversifier les ventes et ont été abordées dans d'autres travaux [Jannach *et al.*, 2013; Szlávik *et al.*, 2011].

### 3.2.2 Les méthodes de diversification

Dans cette partie, nous allons parler des différentes méthodes pour produire des listes de recommandations diversifiées. Pour proposer de la diversité, le système doit trouver un compromis entre la similarité et la diversité [Smyth et McClave, 2001], les préférences et la diversité [Zhang et Hurley, 2008], ou la popularité et la précision [Celma et Cano, 2008]. La sélection d'articles divers est un problème NP-difficile. Une solution est d'utiliser une approche gloutonne en maximisant une fonction objectif submodulaire [Ashkan *et al.*, 2015; Carbonell et Goldstein, 1998]. Le gain que l'on obtient en ajoutant un élément à la liste de recommandation peut être estimée de manière implicite ou explicite que nous verrons ensuite. Dans le processus implicite, le but est de pénaliser les articles trop semblables aux articles se trouvant déjà dans la liste. Cette méthode maximise la distance moyenne entre les paires prises deux à deux [Carbonell et Goldstein, 1998; Parambath *et al.*, 2016]. Le processus explicite cherche à maximiser dans la liste la présence des différents thèmes qui intéressent l'utilisateur, tout en évitant les thèmes

redondants [Wasilewski et Hurley, 2016]. Nous notons que ces deux techniques qui cherchent à ajouter de la diversité dans les recommandations reclassent les résultats obtenus par un premier système de recommandation. Outre ces deux méthodes, il existe aussi des méthodes qui exploitent les techniques de partitionnement afin de proposer des listes diverses (section 3.2.2.3).

#### 3.2.2.1 Les méthodes implicites

Une des premières méthodes de diversité provient de la recherche d'information [Carbonell et Goldstein, 1998]. Cette méthode est utilisée pour réduire les informations redondantes dans les résultats pertinents retournés par le système. Cette méthode s'appelle *Maximal Marginal Relevance* (MMR), et utilise une approche gloutonne pour créer la liste de recommandation en cherchant un compromis entre la pertinence d'un document et la quantité d'information que ce document apporte par rapport aux documents déjà sélectionnés. Dans les systèmes de recommandation, pour chaque produit, MMR fait la combinaison linéaire de la similarité entre le profil de l'utilisateur et le produit à insérer dans la liste et la similarité maximale entre le produit à insérer et tous les produits déjà sélectionnés (formule 3.9).

$$\text{MMR} = \underset{i \in I \setminus I_u \cup S}{\operatorname{argmax}} \lambda sim_1(u, i) - (1 - \lambda) \max_{j \in S} sim_2(i, j) \tag{3.9}$$

Une autre méthode analogue à MMR a été proposée par Borodin *et al.* [2012], la *Max-Sum Diversification* qui comme MMR est une combinaison linéaire d'une fonction qui mesure la pertinence d'un article pour un utilisateur $g(S)$ et une fonction qui mesure la différence entre les articles sélectionnés (formule 3.10).

$$\text{MSD} = \underset{S \subseteq I \setminus I_u}{\operatorname{argmax}} g(S) - (1 - \lambda) \sum_{i \in S} \sum_{j \in S - \{i\}} Dist(i, j) \; s.t \;\; |S| \leq k \tag{3.10}$$

Ces deux fonctions objectif utilisent le paramètre $\lambda \in [0, 1]$ pour faire le compromis entre diversité et pertinence.

#### 3.2.2.2 Les méthodes explicites

Les méthodes explicites proviennent aussi des systèmes de recherches d'informations. Elles cherchent avant tout à identifier les thèmes et sous thèmes d'une requête en utilisant des méthodes de reformulation de la requête ou en utilisant des catégorisations. Les documents sont sélectionnés en maximisant la couverture des thèmes inférés. La méthode *xQuaD* (formule 3.11) [Santos *et al.*, 2010] a donné de meilleurs résultats que la méthode implicite [Carbonell et Goldstein, 1998].

$$\text{xQuAD}(i|S) = (1 - \lambda)p(i|u) + \lambda p(i, \neg S|u) \tag{3.11}$$

L'adaptation de cette formule aux systèmes de recommandation, peut se faire ainsi : $p(i|u)$

la probabilité que l'article $i$ soit observé connaissant le profil de l'utilisateur et $p(i, \neg S|u)$ la probabilité d'observer l'article $i$ mais non les articles qui sont déjà dans la liste $S$ [Vargas, 2015].

Toujours dans la recherche d'information, Agrawal *et al.* [2009] font l'hypothèse qu'il existe une taxonomie de thèmes ($C$) qui modélise les sous-thèmes possibles $c$ des requêtes de sorte que les documents et les requêtes appartiennent à plusieurs thèmes. Les auteurs font aussi l'hypothèse que des statistiques d'usage ont été collectées sur les thèmes. Connaissant les catégories auxquelles appartiennent les requêtes et les documents, les caractéristiques d'usage permettent de déterminer la probabilité qu'une catégorie appartienne à un document. Ils ont introduit *IA-Select*(*intent aware*), un algorithme glouton qui sélectionne des documents à partir d'une première liste de documents retournés par un algorithme de l'état de l'art. Ici aussi,la formule a été adaptée aux systèmes de recommandation (formule 3.12) [Vargas, 2015].

$$\text{IA-Select}(i|S) = \sum_{c} p(c|u)\hat{r}_{norm}(u,i)p(c|i)\prod_{i' \in S}(1 - p(c|i')\hat{r}_{norm}(u,i')) \tag{3.12}$$

#### 3.2.2.3 Clustering et autres méthodes pour la diversification

Certaines méthodes utilisent les techniques de partitionnement (*clustering*) pour apporter de la diversité dans les systèmes de recommandation. Des méthodes de filtrage collaboratif ont été détournées dans le but de produire des recommandations diverses [Boim *et al.*, 2011; Li et Murata, 2012]. Boim *et al.* [2011] ont identifié des articles représentatifs des partitions de manière à ce que la distance moyenne entre les articles de la partition et leur « représentant » soit minimisée. Li et Murata [2012] utilisent une méthode de partitionnement multidimensionnelle et sélectionnent des partitions d'articles en utilisant un algorithme des plus proches voisins comme articles candidats à recommander à l'utilisateur actif.

D'autres travaux ont plutôt appliqué la méthode de clustering au profil de l'utilisateur. Dans ces travaux, les auteurs modélisent le profil des utilisateurs afin de connaitre sa propension à apprécier des articles diversifiés [Noia *et al.*, 2017]. Abbassi *et al.* [2009] ont étudié les profils des utilisateurs, et ont créé une méthode qui permet d'aller au-delà de ce profil, en sortant l'utilisateur des sentiers déjà connus. Ils ont aussi identifié des ensembles d'articles peu connus des utilisateurs permettant de faire des découvertes intéressantes. Vargas et Castells [2013] ont identifié la diversité qu'il y avait dans le profil d'un utilisateur et ont généré des recommandations en se basant sur les différents aspects du profil de l'utilisateur. Ils ont combiné ensuite les recommandations pour produire une liste finale.

## 3.3 Perception de la diversité et de la nouveauté par les utilisateurs

Les théories sur la bulle de filtre, la personnalisation accrue, et l'uniformisation de goûts ont poussé la recherche sur les systèmes de recommandation à s'intéresser à des concepts tels que la diversité, la nouveauté et la sérendipité. Des algorithmes ont été proposés pour rendre les recommandations diverses, et enclines à proposer des contenus nouveaux aux utilisateurs. Des études ont été réalisées par la suite avec des utilisateurs pour mesurer l'impact de ces algorithmes sur leur satisfaction. L'algorithme de diversification de thèmes présenté par Ziegler *et al.* [2005] propose de faire un compromis entre la pertinence et la diversité. Une étude sur les utilisateurs a permis de voir que les utilisateurs apprécient un certain degré de diversité dans les listes fournies. La position des articles dans la liste de recommandations a aussi été étudiée [Ge *et al.*, 2010, 2012]. Les auteurs ont constaté que le fait de grouper les articles à fort degré de diversité - les articles peu similaires au profil de l'utilisateur - dans la liste de recommandations réduisait la perception de la diversité par l'utilisateur. Le mieux était de disperser ces types d'articles dans la liste. Sur ce même thème Hu et Pu [2011] ont travaillé sur le rôle de l'interface sur la perception de la diversité. Les résultats ont permis de voir qu'un partitionnement par catégorie était préférable pour favoriser la perception de la diversité. Aussi, Castagnos *et al.* [2013] ont réalisé une expérience sur 250 utilisateurs, et les résultats de cette étude ont permis de confirmer l'impact positif de la diversité sur la satisfaction de l'utilisateur. Ils ont vu que les modèles de préférence divers avaient un impact positif pendant la phase de démarrage à froid. Enfin, ils ont montré que la diversité bien que bénéfique doit être utilisée avec parcimonie, car elle peut susciter la méfiance et l'incompréhension chez l'utilisateur. Les utilisateurs peuvent avoir différents besoins en diversité et nouveauté. Les travaux présentés par Maccatrozzo *et al.* [2017] ont prouvé qu'en fonction du profil et de la curiosité d'un utilisateur, il fallait adapter le degré de diversité et de nouveauté à ajouter dans les recommandations.

Chapitre 4

# Conclusion de l'état de l'art

Dans cette partie, nous avons défini les systèmes de recommandation. Il s'agit d'un ensemble de techniques et de services qui ont pour but de proposer à des utilisateurs des articles susceptibles de les intéresser. Nous avons présenté les différentes techniques employées pour faire des recommandations. Nous avons vu qu'il existe plusieurs techniques et qu'elles peuvent être classées en trois grandes catégories : le filtrage collaboratif, le filtrage basé sur le contenu et une méthode hybride qui combine les deux premières méthodes.

Nous avons ensuite dégagé des problématiques dans la recherche sur les systèmes de recommandation. Nous avons vu qu'une bonne recommandation ne dépend pas seulement de l'aptitude que le système a à prédire les évaluations passées des utilisateurs. Nous avons aussi vu que les systèmes prédictifs précis, les systèmes qui se basaient seulement sur la similarité, ou seulement sur la popularité, n'étaient pas ceux qui apportaient le plus d'utilité aux utilisateurs. Il était nécessaire d'explorer d'autres dimensions autres que la précision, la similarité et la popularité pour continuer à proposer de nouvelles formes de recommandation.

Dans cette optique, nous avons ensuite présenté la nouveauté et la diversité. Ces méthodes ont été produites pour répondre aux problèmes de bulle de filtre, de personnalisation accrue et pour améliorer l'expérience des utilisateurs. Ces méthodes sont aussi importantes du point de vue de l'entreprise qui utilise le système de recommandation puisqu'elles peuvent lui permettre de faire remonter les articles de la longue traîne, et mieux exposer son catalogue. Elle peut ainsi diversifier ses ventes. Cette stratégie s'avèrera payante aussi pour les artistes ou les articles peu connus qui meurent dans la longue traîne.

Dans la communauté des systèmes de recommandation, depuis les articles de Smyth et McClave [2001], de Ziegler *et al.* [2005] et de McNee *et al.* [2006] les équipes de chercheurs ainsi que les entreprises (*Spotify* et son service *Discovery Weekly*, *Deezer* et son service *My Flow*, *1D touch et Divercities* et leurs algorithmes audacieux, etc.) se sont rendus compte que les systèmes de recommandations doivent aussi prendre en compte la nouveauté, la diversité et la sérendipité. Nous pensons qu'il est difficile de faire progresser le champ de recherche sans prendre en compte ces dimensions. Il existe plusieurs méthodes qui définissent et favorisent la nouveauté et la diversité. Dans la suite de la thèse, nous allons présenter nos méthodes qui permettent de créer des listes de recommandations diverses. Nous sommes partis de l'hypothèse qu'il existe une zone propice à la découverte qui se trouve à une certaine distance du profil de l'utilisateur. Nous avons défini cette zone et nous avons proposé des listes de recommandations qui ont l'avantage de maximiser la diversité, la nouveauté sans une grande perte en précision.

Deuxième partie

# Contribution : des recommandations audacieuses

CHAPITRE 5

# SIMILARITÉ ET DISSIMILARITÉ POUR DES DONNÉES FAIBLEMENT DÉCRITES

*a defo chen, kabrit al lachas*

Proverbe haïtien

## Difficulté d'établir des mesures de similarité et de dissimilarité

Comme nous l'avons vu dans les chapitres précédents, la similarité est une notion essentielle dans certaines techniques de recommandation. Dans le filtrage collaboratif, on retrouve la similarité ou la dissimilarité dans les méthodes de classification de type *kNN*. Dans les techniques de recommandation dites basées sur le contenu, on retrouve la notion de similarité lors de la phase de comparaison des attributs des articles à ceux des utilisateurs. La mesure de similarité, fondamentalement, permet de répondre à la question suivante : « est-ce que deux entités se ressemblent ? ». La réponse à cette question sera donnée par fonction mathématique qui va comparer les entités et indiquer le degré de proximité par un scalaire.

Le champ applicatif des mesures de similarité et de dissimilarité ne se restreint pas aux systèmes de recommandations. À titre d'exemple, nous les retrouvons dans les systèmes de recherche d'information avec des problématiques d'indexation de documents [Huang, 2008], dans la comparaison de séquences d'ADN pour la bio-informatique [Pesquita, 2017], dans l'ingénierie des connaissances avec les problématiques d'alignement d'ontologie [Jain *et al.*, 2010]. La similarité ou la dissimilarité entre les entités se fait en comparant leurs descripteurs respectifs. Ils peuvent être du texte (description, biographie, résumé), des métadonnées (acteurs, artistes, producteurs), des mots-clés (genres), des données spatiales, des séquences de caractères. Ces informations descriptives peuvent être fournies avec les entités à comparer. À titre d'exemple, un système d'indexation d'articles scientifiques peut exploiter les résumés, les noms des auteurs, les mots-clés pour calculer des scores entre les articles. Ces informations descriptives peuvent aussi être extraites automatiquement, surtout pour les données multimédias (p. ex., images, audios, vidéos). Même si des solutions existent, l'extraction de caractéristiques de données multimédia est moins aisée que dans le cadre des données textuelles. En outre, la similarité

à défaut d'avoir un chien, nous emmenons notre chèvre à la chasse

ou la dissimilarité peut aussi se faire en exploitant les relations explicites qui existent entre les entités dans un graphe, par exemple les concepts dans une ontologie décrite par une hiérarchie de subsomption où un thésaurus ou les termes sont reliés entre eux par des relations de synonymie, de hiérarchie et d'association.

Dans le cas des systèmes de recommandation, généralement le calcul de similarité se fait en se basant sur les interactions que les utilisateurs ont eues avec les articles. En utilisant la dualité de la matrice d'utilité tirée de ces interactions, il y a la possibilité de retrouver les utilisateurs similaires, mais aussi les articles similaires. L'indice de similarité entre les utilisateurs peut présager du fait que les utilisateurs partagent des caractéristiques similaires, au moins au niveau du goût. En effet, deux utilisateurs similaires auront eu dans le passé un comportement très proche sur la plateforme. Par ailleurs, nous ne pouvons pas nous risquer à faire une supposition semblable pour l'indice de similarité entre les articles. La similarité entre les articles dépend des habitudes de consommation des utilisateurs. Plusieurs utilisateurs peuvent apprécier et consommer les mêmes séquences de films, ou de musiques sans pour autant que ces articles ne partagent les mêmes caractéristiques (p. ex., films de super héros et films d'auteur ; rock des années 70 et *deep house*). Les goûts éclectiques de ces utilisateurs permettront de faire le rapprochement entre des articles qui ne partagent pas à priori des caractéristiques semblables.

Par ailleurs, il est très difficile de détecter la similarité entre les articles et les utilisateurs du fait du nombre limité d'informations dans la matrice d'utilité. Ce phénomène est accentué quand le système de recommandation propose des articles de niches, tels que des artistes indépendants, où la communauté active est moins importante que celle qui consomme les artistes *mainstream*, soutenus par des grandes compagnies de l'industrie du disque (les « *majors* »). Nous retrouvons aussi ce problème de pénurie de données au niveau des données descripteurs de ces artistes. Le catalogue de ces artistes est très peu décrit. Les descripteurs sont souvent très pauvres (genre musical) ou parfois absents. Le challenge est alors de retrouver des sources et des ressources sur le *web* pour enrichir ces données. Ce processus consiste à améliorer, affiner un objet à partir de sources de données hétérogènes. Sur le *web*, les sources de données sont caractérisées par la structure des données. Elles peuvent être entièrement structurées, semi-structurées et non structurées [Gomadam *et al.*, 2012; Park et Widom, 2014]. Le processus d'extraction d'attributs d'objets manquants directement des pages *web html* s'effectue via des programmes appelés *wrappers* [Furche *et al.*, 2014]. D'un autre côté, il existe des procédés bien plus simples où il suffit de requêter des *API* publiques pour accéder à des informations sur les données culturelles. Nous pouvons citer à titre d'exemple *Acousticbrainz* [1] ou *Discogs* [2] pour la musique. Les données récupérées sur ces *API* (c.-à-d. une interface de programmation applicative) sont dans des formats lisibles facilement par les machines (p. ex., *JSON*, *XML*). Ces informations sont le plus souvent données par des experts ou par une communauté modérée d'amateur d'art. La connaissance

1. https://acousticbrainz.org/data
2. https://www.discogs.com/

peut également être extraite de données encyclopédiques (p. ex., *Wikipédia* [3]) ou des sources de données ouvertes liées, des bases dites sémantiques (p. ex., *DBpedia* [4], *BBC Musique* [5]).

Somme toute, pour arriver à mettre en place un système de recommandation basé sur le contenu, il nous faut une mesure de similarité pertinente entre nos deux ensembles d'entités (c.-à-d. utilisateurs et articles). Mais la pertinence de notre mesure dépend grandement de la qualité des descripteurs des articles. Or, nous travaillons avec des données provenant d'artistes indépendants. Les artistes indépendants souffrent d'un manque de visibilité sur le web dû aussi à la pauvreté de la qualification de leurs données. Il est plus aisé d'obtenir des informations *people* concernant Eminem que les informations sur le groupe stéphanois Altam. Les données des artistes indépendants sont peu qualifiées. Donc nous avons adopté une stratégie de récupération de données sur des sources externes et nous avons pu appliquer des mesures de distance et de similarité classique, ainsi que des mesures de distance et de similarité que nous avons adaptées à de situations de pénuries de descripteurs.

Dans la suite de ce chapitre, nous présenterons les méthodes que nous avons développées et qui utilisent des données provenant de bases sémantiques, des données provenant d'*API* publiques et des données provenant d'une folksonomie récupérée sur *Last.fm*. Dans la Section 5.1, nous présentons les méthodes de calculs de similarité développées à partir des données sémantiques et des *tags* d'une folksonomie. Nous présenterons ensuite dans la Section 5.2 les méthodes de représentations vectorielles que nous avons utilisées pour améliorer la précision des mesures de similarité.

Comme nous l'avons vu, les sources de données sont nombreuses. En revanche, les artistes indépendants souffrent d'un manque de référencement dans ces bases de données. Ils sont très peu présents sur le *web*. Dans la Section 5.3, nous décrirons une méthode qui permet d'obtenir un indice de similarité pertinent pour les artistes faiblement décrits.

## 5.1 Données sémantiques et Folksonomie pour une mesure de similarité et de dissimilarité

### 5.1.1 Les Folksonomies

Lorsque l'on souhaite réaliser une application telle qu'un système de recommandation basé sur le contenu, il faut essayer de disposer de données de qualité, structurées, et si possible présentes en grand nombre. Certains sites se basent sur des systèmes d'étiquetage (*tagging system*) appelés aussi « folksonomie » qui permettent à des utilisateurs non spécialistes d'annoter et d'indexer publiquement et de manière spontanée des articles. Cette pratique est supportée par

3. https://fr.wikipedia.org/wiki/Brain_Damage_(groupe_de_dub)
4. http://fr.dbpedia.org/page/Brain_Damage_(groupe_de_dub)
5. https://www.bbc.co.uk/music/artists/83d91898-7763-47d7-b03b-b92132375c47

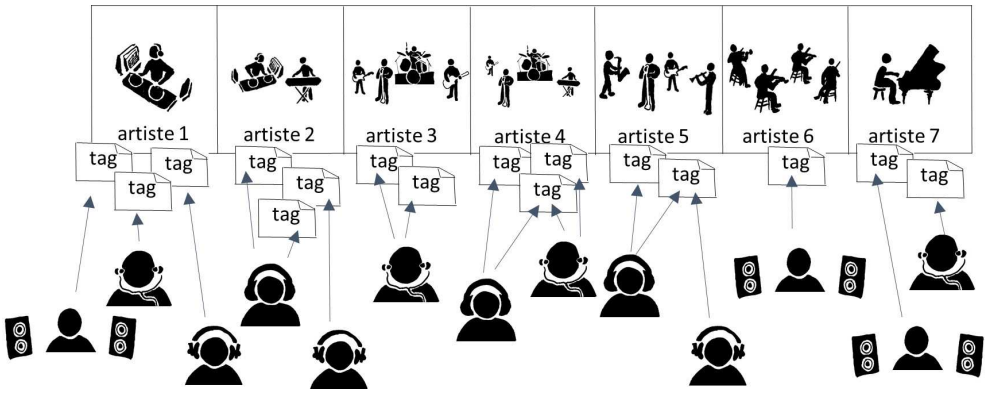



FIGURE 5.1 – Production d'une folksonomie : les utilisateurs de la plate-forme musicale annotent les artistes musicaux par des tags.

des sites comme *Delicio.us* [6], *Technorati* [7], *Flickr* [8], *Last.fm* [9]. Les utilisateurs peuvent annoter des ressources incluant des images, des artistes musicaux, des pages *web*, des vidéos, etc. Les annotations d'une folsonomie ne sont pas contraintes d'appartenir à une terminologie prédéfinie telle qu'une ontologie. Selon les auteurs, ces annotations sont appelées « mots-clés », « étiquettes » ou plus simplement « tags », et la connaissance qu'elles portent permet de caractériser les articles selon l'humeur et les sentiments qu'il procure, des opinions personnelles ou selon des caractéristiques plus factuelles telles que le genre, la décennie, la nationalité, etc. Dans le jeu de données de *Last.fm* [Cantador *et al.*, 2011], on retrouve à titre d'exemple des « tags » tels que (i) *atmospheric*, (ii) *german*, (iii) *cafe del mar*, (iv) *classic 80s*, (v) *best singer ever*. pour qualifier des artistes musicaux. Les utilisateurs bénéficient d'une liberté totale au niveau du choix des « tags ». Cette liberté soulève des difficultés au niveau de traitement du langage naturel. Pour un même sens, on retrouve des déclinaisons dans plusieurs langues (p. ex., *french pop* et *pop française*), des synonymes, plusieurs formes (p. ex., pop rock, pop-rock), etc.

Sur la figure 5.1, nous illustrons la manière dont une folksonomie peut être produite dans un système d'écoute musicale : les différents utilisateurs du système attribuent des tags aux différents produits (ici des artistes musicaux) en fonction de leurs impressions et appréciations spontanées.

En règle générale, une folksonomie est définie par un tuple $F = (T, U, I, A)$ ou $T$ est l'ensemble des « tags », $U$ est l'ensemble des utilisateurs, $I$ l'ensemble des ressources et $A$ l'ensemble des annotations, ou $A = (u, t, i) \in U \times T \times I$.

Même si, dans quelques situations, les tags d'une folksonomie peuvent refléter un degré d'expertise des utilisateurs contribuant au système d'annotations, par exemple une certaine expertise dans le contexte des communications d'entreprise [John et Seligmann, 2006], les tags sont le plus souvent attribués par des utilisateurs non experts.

6. https://del.icio.us/
7. http://technorati.com/
8. https://www.flickr.com/
9. https://www.last.fm/fr/

### 5.1.2 Les données structurées

Il existe par ailleurs des approches qui mettent en avant les informations expertes, structurées, liées en graphe, issues des technologies sémantiques. Ces technologies associées à la disponibilité de plusieurs sources de connaissances ouvertes ont permis d'accomplir de grands progrès en étant adaptées aux systèmes de recommandation basés sur le contenu [Gemmis *et al.*, 2015a]. La découverte de caractéristiques porteuses de sens dans un texte est fondamentale pour son traitement, ainsi l'intégration de sources de connaissances externes est précieuse pour ajouter du contenu et améliorer la représentation des articles, apportant une connaissance linguistique ainsi que des éléments de connaissance en lien avec le domaine. L'ensemble de cette connaissance peut être fournie par diffèrentes sources, incluant des ontologies, des données encyclopédiques non structurées (p. ex., *Wikipédia*) ou des sources de données ouvertes liées en ligne (p. ex., *DBpedia*). Le format des données ouvertes et liées (*web* des données) semble prometteur : les données liées permettent d'enrichir les descriptions des articles en exploitant les différentes caractéristiques du réseau sémantique.

Dans ce qui suit, nous montrons l'utilisation des données récupérées pour le calcul de similarité et de dissimilarité en vue de les utiliser dans des systèmes de recommandation basés sur le contenu. Nous montrons aussi que les données sémantiques et les « tags » issus d'une folksonomie peuvent s'associer et s'enrichir mutuellement.

## 5.2 Des représentations vectorielles pour des mesures de similarité

Dans cette section, nous allons présenter les différentes approches que nous avons utilisées pour calculer la similarité entre des artistes musicaux. Nous avons essayé plusieurs représentations vectorielles issues de la littérature afin de choisir celles qui nous conviendraient le mieux dans un système de recommandation basé sur le contenu.

#### Notation

Nous considérons un ensemble d'articles (ici des artistes musicaux) que nous notons $I = \{i_1, ..., i_n\}$. Le but est de trouver l'indice de similarité $sim(i, j)$ et de dissimilarité $Dist(i, j)$ entre chaque paire d'articles appartenant à $I$ avec $s \in [0, 1]$, et $d = 1 - s$, avec $s = 1$ pour $i = j$ et $d = 0$ pour $i = j$ en se basant sur les « tags » $T$ de la folksonomie et le contenu des données structurées.

### 5.2.1 Encodage *One-hot* et par fréquence

#### 5.2.1.1 Similarité déduite de la folksonomie

Un premier mode de calcul de la similarité entre artistes peut se faire à partir des tags en faisant l'hypothèse que plus les artistes sont décrits par des tags identiques, plus ils sont considérés comme étant similaires.

Pour chaque artiste $i$ on peut retrouver l'ensemble des « tags » $T_i$ qui ont servi à l'annoter. Nous faisons une représentation vectorielle binaire de chaque artiste $i$ telle que :

$$t = \begin{cases} 1 & \text{if } t \in T_i \\ 0 & \text{if } n \notin T_i \end{cases}$$

Nous utilisons l'indice de Jaccard adapté pour trouver la similarité entre des objets constitués d'attributs binaires. Il s'agit du rapport entre le cardinal de l'intersection des ensembles et le cardinal de l'union des ensembles. Nous avons la similarité de Jaccard donnée par la formule 5.1 et la distance de Jaccard donnée par la formule 5.2.

$$\text{sim}_{\text{jaccard}}(i, j) = \frac{|i \cap j|}{|i \cup j|} \tag{5.1}$$

$$\text{Dist}_{\text{jaccard}}(i, j) = 1 - s(i, j) \tag{5.2}$$

Nous faisons aussi une représentation sac de mots où le « tag » est représenté par sa fréquence dans le vecteur de représentation de l'artiste. Nous utilisons la distance euclidienne plus adaptée aux vecteurs composés de valeurs continues (formule 5.3). La similarité euclidienne est donnée par la formule 5.4.

$$\text{Dist}_{\text{euclidienne}}(i, j) = \sqrt{\sum_{k=1}^{d} (i_k - j_k)^2} \tag{5.3}$$

$$\text{sim}_{\text{euclidienne}}(i, j) = \frac{1}{1 + d(i, j)} \tag{5.4}$$

#### 5.2.1.2 Similarité déduite de données sémantiques

Nous avons utilisé les données du graphe sémantique pour établir un second mode de calcul de la similarité entre artistes. Notre approche consiste à apparier les artistes musicaux d'une plate-forme musicale avec leur page correspondante sur le site de données sémantiques en ligne *DBpedia*, un site présentant les informations structurées issues de *Wikipédia* de manière à rendre ces pages exploitables par les approches du *web* sémantique. Au sein de ces pages sont récupérées les informations jugées pertinentes pour la description des artistes, telles que le résumé biographique ("*abstract*"), les genres musicaux ("*genre*"), les maisons de disque ("*labels*"), etc. Nous n'utilisons pas la structure de graphe de données de *DBpedia*. Nous nous

limitons aux informations textuelles et nous réalisons sur ces textes le même traitement que sur les « tags ».

Nous faisons une représentation binaire des informations extraites de *DBpedia*. Nous calculons une similarité et une dissimilarité entre les artistes en utilisant les Formules 5.1 et 5.2 en fonction des biographies, des genres, des maisons de disque, etc.

Nous optons par la suite pour une représentation sac de mots pour les biographies. Nous créons la matrice artistes/termes où les cases de la matrice sont les fréquences d'apparition des termes dans les biographies des artistes. Nous utilisons les Formules 5.4 et 5.3 pour calculer la similarité et la distance entre deux artistes. Nous ne faisons pas de représentation sac de mots pour les autres informations. Un genre, un artiste, une maison de disque auront toujours une fréquence binaire pour chaque artiste.

À partir de toutes ces similarités et ces distances, nous déduisons une distance et une similarité globale entre chaque pair d'artistes donnée par les formules suivantes, 5.5 et 5.6 :

$$\mathrm{s}(i,j) = \alpha_{genres} \times s_{genres}(i,j) + \alpha_{abstract} \times s_{abstract} + \alpha_{label} \times s_{label} + ... \tag{5.5}$$

$$\mathrm{d}(i,j) = \alpha_{genres} \times d_{genres}(i,j) + \alpha_{abstract} \times d_{abstract} + \alpha_{label} \times d_{label} + ... \tag{5.6}$$

Le paramètre $\alpha$ permet de pondérer les différentes similarité et dissimilarité.

### 5.2.2 Techniques de réduction de dimensionnalité

Nous avons testé plusieurs méthodes de réduction de dimensionnalité. Le but est de réduire les vecteurs de représentation des artistes afin d'améliorer le résultat des calculs de proximité entre les éléments de notre base de données. Cette étape est importante pour le développement des systèmes de recommandation que nous proposerons. Ils permettront aussi d'affiner des profils d'utilisateurs et aussi de prédire les futurs scores d'écoute de ces utilisateurs. Nous avons utilisé les techniques d'indexation sémantique : l'analyse sémantique latente, l'allocation latente de dirichlet, et les techniques de plongement lexical, Word2Vec et GloVe.

#### 5.2.2.1 Les techniques d'indexation sémantique

Nous avons utilisé des méthodes d'indexation sémantique pour obtenir des vecteurs de représentation des artistes en utilisant les données de la folksnomie et du web sémantique. Pour les deux types de données, nous avons créé des matrices termes × artistes. À noter que pour les informations issues des données structurées, nous avons créé des matrices pour deux types de données récupérées, les biographies et les genres musicaux. Nous avons par la suite utilisé *TF-IDF* (*Term-frequency inverse-document-frequency*) pour pondérer les matrices créées. Chaque élément des matrices est calculé en multipliant la fréquence du terme dans le document par

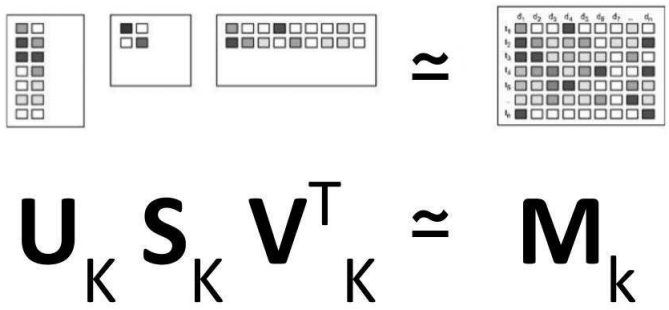



FIGURE 5.2 – Le modèle LSI.

l'inverse de la proportion de documents du corpus qui contiennent le terme (formules 5.7 et 5.8).

$$\text{idf}(t, D) = log\left(\frac{|D|}{|d \in D : t \in d|}\right) \tag{5.7}$$

$$\text{tfidf}(t, D) = tf_{t,D} \times idf_t \tag{5.8}$$

**5.2.2.1.1 Indexation Sémantique Latente (*LSI*)** Nous avons réduit chaque matrice en utilisant l'indexation sémantique latente *Latent Semantic Indexing* [Deerwester *et al.*, 1990]. Cette méthode permet de retrouver le sens latent dans les documents en créant des concepts à partir des termes. Elle fait l'hypothèse que des mots qui apparaissent dans un même contexte sont sémantiquement proches. Le contexte peut être un document, un paragraphe, une phrase, etc.

Elle fait usage d'une décomposition en valeur singulière (*SVD*) sur la matrice terme × document pour obtenir une version compressée de cette matrice originale. Cette matrice se décompose en trois matrices $USV^T$ avec $U$ et $V$ des matrices orthogonales et $S$ une matrice diagonale, comme cela est présenté en figure 5.2 [10]. Considérons $k, 0 < k < r$, on enlève de la matrice $S$ les $r - k$ colonnes qui ont les plus petites valeurs singulières. On enlève ensuite des matrices $U$ et $V$ les colonnes correspondantes. On obtient une matrice réduite de rang $k$ $U_k S_k V_k^T$.

**5.2.2.1.2 L'Allocation Latente de Dirichlet (*LDA*)** Nous avons fait usage de l'Allocation Latente de Dirichlet (*LDA*) [Blei *et al.*, 2001] afin de réduire la dimension des matrices (pour la méthode *LDA* nous avons utilisé les matrices TF). Il s'agit d'un modèle probabilistique qui permet de retrouver la structure des thèmes cachés dans un corpus de document. Il repose sur un modèle bayésien hiérarchique à trois couches ou chaque document est modélisé par un mélange de thèmes qui génère chaque mot du document. Nous obtenons des vecteurs de dimensions fixes dont chaque dimension correspond à la probabilité que le document (artiste) appartienne au

10. `https://pal.cct.brookes.ac.uk/linear-algebra-latent-semantic-analysis-lsa/`

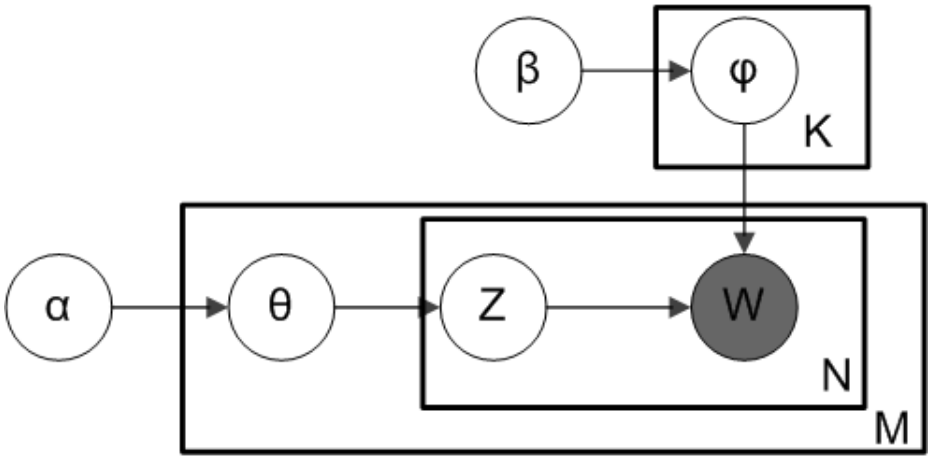



FIGURE 5.3 – Le modèle génératif LDA de Blei *et al.* [2001].

thème en question.

Sur la figure 5.3, nous présentons le modèle LDA avec les noeuds $\alpha$ et $\beta$ représentant les paramètres de deux distributions de Dirichlet. Les noeuds $\theta$ et $\phi$ représentent les paramètres de deux distributions multinomiales. Sur ce schéma, chaque thème $z_n$ indexe une distribution sur les mots dans le thème, et chaque $z_n$ est tiré de la distribution multinomiale $\theta$.

#### 5.2.2.2 Le modèle Word2Vec

*Word2vec* [Mikolov *et al.*, 2013] est une technique de plongement lexical (*word embedding*) qui permet de représenter chaque mot par un vecteur de nombres réels. Le modèle permet de représenter les mots dans des espaces vectoriels de rang réduits en se basant sur la distribution de ces mots dans un corpus de texte donné. *Word2vec* utilise deux modèles, le *CBOW* et le *skip-gram*, qui sont deux modèles de réseaux de neurones peu profonds. Il s'agit de deux modèles peu complexes, mais qui deviennent performants s'ils utilisent de grands jeux de données en phase d'apprentissage. Le modèle *CBOW* est entrainé pour prédire le terme $t$ sachant le contexte (c.-à-d., les mots qui l'entourent). Le but est de maximiser $P(t|C)$. Le modèle *skip-gram* fait l'inverse, il cherche à prédire le contexte en sachant le terme. Sur la figure 5.4 [11], nous avons une représentation graphique des modèles *CBOW* et *skip-gram*

Dans nos expérimentations, nous avons utilisé le modèle *CBOW* pour apprendre des vecteurs de représentation des termes des biographies des artistes. Pour obtenir un vecteur unique de représentation d'un artiste, nous additionnons tous les vecteurs des mots de sa biographie. Nous ne normalisons pas les vecteurs. Nous utilisons la distance du cosinus pour obtenir un indice de dissimilarité entre les artistes.

11. `https://deeplearning4j.org/word2vec`

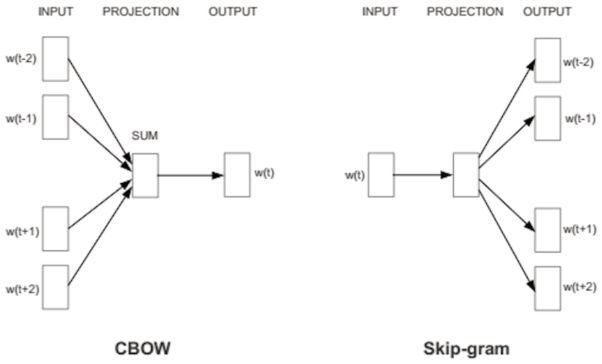



FIGURE 5.4 – Les modèles Word2vec de Mikolov *et al.* [2013].

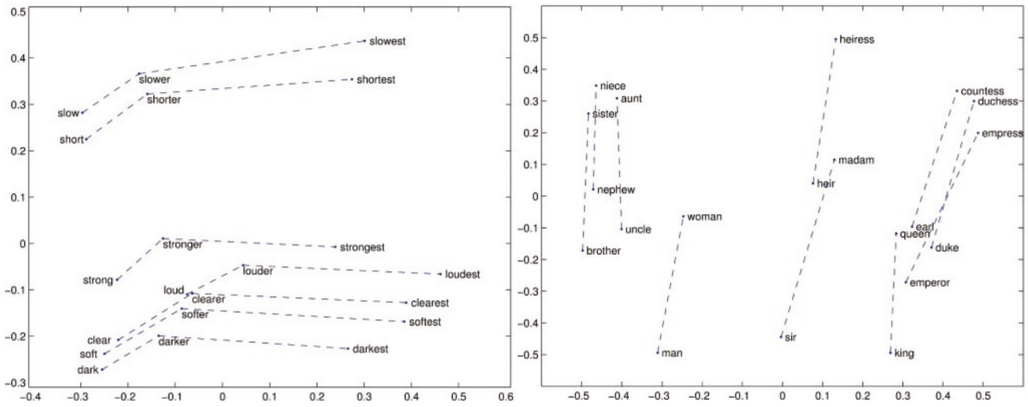



FIGURE 5.5 – Des vecteurs de mots obtenus par le modèle GloVe de Pennington *et al.* [2014].

#### 5.2.2.3 Le modèle GloVe

GloVe (*Global Vectors for word representation*) [Pennington *et al.*, 2014] est une méthode d'apprentissage non supervisée qui se base sur les statistiques globales de co-occurence des mots dans des corpus de grandes tailles pour apprendre des vecteurs de représentation pour ces mots. Nous avons utilisé dans nos expérimentations les vecteurs qui ont été entrainés sur un corpus d'articles de *Wikipedia*. Ces vecteurs peuvent être récupérés sur le site *web* de GloVe (figure 5.5 [12]).

Le modèle GloVe combine les informations de cooccurrence des mots avec les bénéfices des méthodes de factorisation de matrice.

Pour obtenir un vecteur unique de représentation d'un artiste, nous utilisons la même méthode que celle utilisée pour le modèle *Word2vec*. Nous additionnons tous les vecteurs des mots de sa biographie. Nous ne normalisons pas les vecteurs. Nous utilisons la distance du cosinus pour obtenir un indice de dissimilarité entre les artistes.

12. `https://nlp.stanford.edu/projects/glove/`

### 5.2.3 Expérimentations et résultats

#### 5.2.3.1 Jeux de données

Nous avons réalisé des expériences sur un jeu de données fourni par Oramas *et al.* [2015] du *Music Technology Group* de l'Université Pompeu Fabra à Barcelone (Espagne). Il s'agit d'un jeu de données permettant d'évaluer des mesures de similarité entre des artistes. Il contient deux corpus d'artistes, un corpus de 268 artistes et un autre plus grand de 2 336 artistes récupérés sur *Last.fm*. Le premier corpus de 268 artistes a été jumelé au jeu de données *MIREX Audio and Music Similarity evaluation dataset* afin d'utiliser le jugement de similarité de ce dernier comme vérité terrain. Le jeu de données *MIREX Audio Music Similarity task dataset* contient 7 000 musiques de 602 artistes différents et un indice de similarité entre les musiques. La similarité entre deux artistes a été obtenue en calculant la similarité entre leurs morceaux de musique. Deux artistes sont considérés comme similaires si le score de similarité obtenu était supérieur à 25 sur une échelle de 0 à 100. Après avoir nettoyé le corpus, ils ont obtenu une liste de 268 artistes. Chaque artiste a au plus 10 artistes les plus similaires [Oramas *et al.*, 2015]. Nous avons utilisé dans toutes les expériences qui suivent le seul corpus de 268 artistes qui nous a été fourni par Sergio Oramas.

Dans le corpus, les artistes sont identifiés par leur nom, leur *uri (Uniform Resource Identifier)* sur *DBpedia*, ainsi que leur identifiant unique *MusicBrainz*. Des biographies provenant de *Last.fm* sont aussi fournies dans le jeu de données. L'*uri (Uniform Resource Identifier) DBpedia* nous permet d'aller récupérer d'autres informations telles que les genres musicaux, la biographie, le pays d'origine, les musiciens, les récompenses, etc.

Nous avons étendu le jeu de données en allant récupérer les genres musicaux sur *DBpedia*. Nous avons récupéré 215 genres musicaux uniques. Nous avons aussi récupéré les annotations faites par les utilisateurs sur *Last.fm*, soit une liste de 1 388 tags.

#### 5.2.3.2 Les méthodes évaluées

Nous avons appliqué plusieurs méthodes de représentations de la connaissance, de mesures de distance, et de paramétrage. Nous pouvons les classer en deux grandes catégories. Une première qui consiste à représenter les artistes par des vecteurs de fréquences d'apparitions des termes dans les descripteurs que nous normalisons en utilisant la formule du *TF-IDF* (formule 5.8). Les descripteurs sont les *tags*, les genres, les biographies. La taille des vecteurs de représentation sera la taille du vocabulaire (c.-à-d., le nombre de tags, le nombre de genres, le nombre de termes du corpus des biographies). La deuxième catégorie contient les méthodes de réduction de dimensionnalité des vecteurs de représentations des artistes ainsi que le choix des paramètres tels que la taille de ces vecteurs : *LSI*, *LDA*, *Word2vec*, *GloVe* (section 5.2.2.3).

Nous utilisons des formes de nommage pour différencier les modèles créés. Nous différencions le « + » (par exemple bios + genres), du « et » (ex. tags et bios) pour indiquer la manière dont nous avons utilisé les descripteurs. Pour faire allusion aux méthodes « *et* », nous utilisons

le préfixe OV (*One Vector*) devant les méthodes de réduction de similarité et *TF-IDF*. Cela veut dire que les descripteurs (*tags*, genres, biographies) ont été utilisés pour créer un seul vecteur de représentation pour chaque artiste avant d'être réduits. Pour désigner l'autre famille d'utilisation des descripteurs, nous utilisons le suffixe « + », par exemple, *TF-IDF+*. Cela signifie que dans un premier temps, nous avons calculé la distance pour chaque descripteur et, dans un second temps, additionné les distances pour obtenir une distance globale, formule 5.6. Les suffixes *T (trained)* et *PT (pre-trained)* utilisés dans notre nommage des méthodes *word2vec (w2v)* et *GloVe* signifie respectivement, que nous avons entrainé le modèle sur nos données pour obtenir des représentions vectorielles pour les termes, et que nous avons utilisés les vecteurs pré-entrainés disponibles sur le *web*.

#### 5.2.3.3 Méthodologie d'évaluation

Pour valider nos approches et les comparer, nous avons pour chaque artiste source classé les autres artistes selon leur valeur de similarité qu'ils ont obtenue par rapport à l'artiste source. Nous avons sélectionné ensuite dans ces listes les $k$ artistes les plus semblables et nous nous sommes servis de la vérité terrain pour mesurer les performances des différentes méthodes. Nous avons utilisé les mesures de performances standards, la Précision@k, le Rappel@k et la mesure nDCG@k (*normalized discounted cumulative gain*).

| Paramètres | Explication |
|---|---|
| i, k | indices |
| I | ensemble des artistes |
| R | ensemble des résultats retournés |
| T | ensemble test, vérité terrain |
| rel | indice de pertinence |
| $log_2$ | logarithme base 2 |

Tableau 5.1 – Notation et paramètres utilisés dans le mesures d'évaluation.

**5.2.3.3.1 Precision@k** La précision correspond à la quantité d'artistes pertinents $R_i$ retrouvé dans la liste top-k par rapport à la vérité terrain $T_i$

$$\frac{1}{|I|}\sum_{i\in I}\frac{1}{k}|R_i \cap T_i|$$

**5.2.3.3.2 Rappel@k** Il s'agit du rapport d'artistes pertinent retourné dans la liste top-k par la longueur k de cette liste

$$\frac{1}{|I|}\sum_{i\in I}\frac{|R_i \cap T_i|}{T_i}$$

**5.2.3.3.3 Normalized Distributed Cumulative Gain@k (nDCG)** nDCG prend en compte la pertinence et le rang de l'artiste retourné dans la liste top-k. nDCG donnera un plus grand poids aux artistes pertinent positionné plus haut dans la liste. Cette métrique est calculée ainsi, avec $rel_i$ représentant la pertinence de l'artiste $i$ dans la liste de similarité :

$$\frac{1}{|U|} \cdot \sum_{u \in U} \frac{1}{iDCG@N} \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(1+i)}$$

$iDCG@k$ représente le résultat du classement idéal d'un $DCG@k$. Les valeurs possibles de $rel_i$ sont 0 et 1, 0 si l'artiste n'est pas dans la liste « vérité terrain », et 1 s'il s'y trouve.

### 5.2.3.4 Résultats

Nous avons évalué les méthodes décrites sur le jeu de données présenté dans la section 5.2.3.1. Nous présentons pour chaque méthode les résultats obtenus quand la liste d'éléments semblables renvoyée contient 5 artistes, et quand cette liste contient 10 artistes. Nous présentons pour chaque méthode (TF-IDF, LSI, LDA), les descripteurs utilisés, la manière dont les descripteurs sont utilisés, la dimension du vecteur de représentation de l'artiste et les résultats en précision, rappel et nDCG.

Nous présentons d'abord les résultats des méthodes qui font la sommation des distances obtenues à partir des descripteurs (les méthodes « + »). Le tableau 5.2 présente les résultats obtenus par la méthode TF-IDF+. Les résultats de cette méthode montrent que l'utilisation des tags comme éléments de description des artistes permet d'obtenir les meilleurs résultats. L'utilisation des genres permet d'obtenir les deuxièmes meilleurs résultats. L'assimilation des distances obtenues à partir des différents descripteurs n'améliore pas les résultats. Mais l'utilisation de la distance des trois descripteurs pour avoir une distance globale (bios + genres + tags) se révèle plus efficace que l'utilisation de la distance de deux descripteurs (bios + genres).

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags | 1 388 | 5 | **0.237** | **0.166** | **0.317** |
| bios | 14 308 | 5 | 0.070 | 0.051 | 0.097 |
| genres | 215 | 5 | *0.165* | *0.112* | *0.243* |
| bios + genres | - | 5 | 0.070 | 0.051 | 0.097 |
| bios + genres + tags | - | 5 | 0.127 | 0.089 | 0.174 |
| tags | 1 388 | 10 | **0.208** | **0.270** | **0.293** |
| bios | 14 308 | 10 | 0.067 | 0.089 | 0.094 |
| genres | 215 | 10 | *0.133* | *0.160* | *0.206* |
| bios + genres | - | 10 | 0.067 | 0.089 | 0.094 |
| bios + genres + tags | - | 10 | 0.099 | 0.124 | 0.147 |

Tableau 5.2 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode TF-IDF+ sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrain la similarité issue du jeu de données MIREX

Le tableau 5.3 montre les résultats obtenus par la méthode LSI. L'utilisation des tags se révèle ici aussi le plus efficace. L'utilisation des genres pour la méthode LSI obtient les deuxièmes meilleurs résultats. Nous remarquons une nette amélioration des résultats obtenus par la somme des distances des descripteurs (bios + genres, bios + genres + tags), par rapport à la méthode TF-IDF+. La sommation des distances de 3 descripteurs (bios + genres + tags) se révèle encore plus efficace que la sommation des distances de 2 descripteurs (bios + genres).

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags | 10 | 5 | **0.223** | **0.152** | **0.300** |
| bios | 10 | 5 | 0.073 | 0.052 | 0.098 |
| genres | 10 | 5 | *0.191* | *0.127* | *0.263* |
| bios + genres | 5 | 10 | 0.153 | 0.105 | 0.207 |
| bios + genres + tags | 5 | 10 | 0.187 | 0.126 | 0.253 |
| tags | 10 | 10 | **0.205** | **0.256** | **0.286** |
| bios | 10 | 10 | 0.067 | 0.092 | 0.094 |
| genres | 10 | 10 | *0.165* | *0.206* | *0.241* |
| bios + genres | 10 | 10 | 0.129 | 0.162 | 0.185 |
| bios + genres + tags | 10 | 10 | 0.172 | 0.216 | 0.241 |

Tableau 5.3 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode LSI+ sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

Le tableau 5.4 présente les résultats obtenus par la méthode LDA+. LDA+ obtient de moins bons résultats que LSI. L'utilisation des tags permet d'obtenir les meilleurs résultats. En revanche les genres obtiennent sensiblement les mêmes résultats que les tags. La sommation des distances des descripteurs entraine aussi une perte de qualité, mais l'usage des trois distances (bios + genres + tags) atténue cette perte.

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags | 10 | 5 | **0.124** | *0.079* | *0.174* |
| bios | 10 | 5 | 0.038 | 0.028 | 0.051 |
| genres | 10 | 5 | *0.115* | **0.081** | **0.178** |
| bios + genres | 10 | 5 | 0.064 | 0.042 | 0.086 |
| bios + genres + tags | 10 | 5 | 0.105 | 0.066 | 0.140 |
| tags | 10 | 10 | **0.111** | **0.133** | **0.161** |
| bios | 10 | 10 | 0.040 | 0.052 | 0.054 |
| genres | 10 | 10 | *0.101* | *0.125* | *0.158* |
| bios + genres | 10 | 10 | 0.065 | 0.078 | 0.089 |
| bios + genres + tags | 10 | 10 | 0.095 | 0.109 | 0.132 |

Tableau 5.4 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode LDA+ sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

Dans les prochains tableaux que nous présentons, nous rapportons les résultats obtenus par les méthodes « OV » (*One Vector*). Dans cette méthode comme nous l'expliquons plus haut, nous

prenons pour les artistes les vecteurs formés à partir de descripteurs choisis (biographie, genres, tags), ensuite nous listons ces vecteurs pour former un seul vecteur, nous réduisons sa dimension ce qui nous permet de calculer les scores de similarité.

Le tableau 5.5 présente les résultats pour la méthode OV TFIDF. Nous remarquons que les vecteurs qui ont été formés à partir des descripteurs « tags » obtiennent sensiblement les mêmes résultats. L'absence des « tags » fait baisser la précision de la méthode OV TFIDF. Les vecteurs utilisés dans cette méthode ont des dimensions supérieures à 1 000.

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags et bios | 15 696 | 5 | **0.227** | 0.160 | **0.305** |
| tags et genres | 1 603 | 5 | *0.225* | *0.161* | *0.299* |
| genres et bios | 14 523 | 5 | 0.130 | 0.091 | 0.178 |
| bios, genres et tags | 15 911 | 5 | *0.225* | **0.164** | *0.299* |
| tags et bios | 15 696 | 10 | **0.202** | **0.265** | **0.285** |
| tags et genres | 1 603 | 10 | **0.202** | *0.259* | *0.281* |
| genres et bios | 14 523 | 10 | 0.099 | 0.125 | 0.148 |
| bios, genres et tags | 15 911 | 10 | *0.200* | 0.258 | 0.280 |

Tableau 5.5 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode OV TFIDF sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrain la similarité issue du jeu de données MIREX

Le tableau 5.6 met en évidence les résultats de la méthode OV LSI. Nous faisons le même constat que précédemment. L'utilisation des tags dans les vecteurs de représentation permet d'obtenir les meilleurs résultats. Les différences observées entre les résultats des vecteurs qui utilisent les tags sont minimes. La différence se fait au centième. L'écart observé entre les résultats du vecteur formé par les genres et la biographie et ceux des vecteurs utilisant les tags est plus petit que celui observé sur les résultats de la méthode OV TFIDF. La taille des vecteurs est plus petite (dimension = 10).

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags et bios | 10 | 5 | 0.213 | 0.140 | 0.290 |
| tags et genres | 10 | 5 | 0.215 | 0.143 | 0.291 |
| genres et bios | 10 | 5 | 0.165 | 0.109 | 0.222 |
| bios, genres et tags | 10 | 5 | 0.216 | 0.143 | 0.293 |
| tags et bios | 10 | 10 | 0.194 | 0.239 | 0.273 |
| tags et genres | 10 | 10 | 0.191 | 0.232 | 0.271 |
| genres et bios | 10 | 10 | 0.141 | 0.175 | 0.201 |
| bios, genres et tags | 10 | 10 | 0.192 | 0.233 | 0.272 |

Tableau 5.6 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode OV LSI sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

Dans le tableau 5.7, nous montrons les résultats obtenus par la méthode OV LDA. Cette fois, nous observons que le vecteur formé par les biographies, les genres et les tags obtient les

meilleurs résultats. L'écart entre les résultats obtenus grâce à l'usage d'un vecteur formé par ces descripteurs (bios, genres, tags) et ceux obtenus grâce à l'usage d'un vecteur formé des autres données (tags et bios, tags et genres, genres et bios) est un peu plus marqué (4 dixièmes de différence). La dimension des vecteurs est la même que pour OV LSI (dimension = 10).

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| tags et bios | 10 | 5 | 0.079 | 0.054 | 0.124 |
| tags et genres | 10 | 5 | 0.082 | 0.056 | 0.117 |
| genres et bios | 10 | 5 | 0.072 | 0.052 | 0.098 |
| bios, genres et tags | 10 | 5 | 0.116 | 0.085 | 0.169 |
| tags et bios | 10 | 10 | 0.074 | 0.093 | 0.114 |
| tags et genres | 10 | 10 | 0.077 | 0.095 | 0.112 |
| genres et bios | 10 | 10 | 0.064 | 0.083 | 0.091 |
| bios, genres et tags | 10 | 10 | 0.102 | 0.137 | 0.152 |

Tableau 5.7 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour la méthode OV LDA sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

Les tableaux 5.8 et 5.9 nous donnent les résultats pour les méthodes de plongement lexical Word2vec et GloVe. Les deux méthodes utilisent l'information provenant des biographies. La méthode basée sur Word2Vec place les vecteurs dans un espace à 10 dimensions. La méthode basée sur GloVe place les vecteurs dans des espaces à 50, 100 ou 200 dimensions. La différence entre les résultats des deux méthodes n'est pas très grande. Nous remarquons un avantage pour la méthode GloVe. Nous remarquons que la précision obtenue par la méthode basée sur GloVe augmente quand la taille des vecteurs augmente.

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| bios | 10 | 5 | 0.103 | 0.073 | 0.137 |
| bios | 10 | 10 | 0.093 | 0.117 | 0.130 |

Tableau 5.8 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour les méthodes word2vec sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

Dans les tableaux 5.10 et 5.11, nous présentons les résultats obtenus en moyenne par les différentes méthodes. Les méthodes OV sont plus efficaces que les méthodes faisant l'addition des distances. Nous remarquons un léger avantage pour la méthode OV LSI, qui arrive à obtenir de meilleurs résultats dans l'ensemble que la méthode OV TFIDF.

#### 5.2.3.5 Discussion

Dans l'ensemble, la méthode qui obtient en moyenne les meilleurs résultats dans nos expérimentations est la méthode *OVS-LSI*. Nous remarquons que l'utilisation des *tags* dans toutes les méthodes permet de mieux capturer la similarité entre les artistes. Ceci peut être expliqué

| données utilisées | dimensions des vecteurs | k | précision | rappel | nDCG |
|---|---|---|---|---|---|
| bios | 50 | 5 | 0.108 | 0.075 | 0.144 |
| bios | 100 | 5 | 0.113 | 0.073 | 0.149 |
| bios | 200 | 5 | 0.120 | 0.082 | 0.159 |
| bios | 50 | 10 | 0.093 | 0.120 | 0.131 |
| bios | 100 | 10 | 0.100 | 0.124 | 0.139 |
| bios | 200 | 10 | 0.108 | 0.138 | 0.150 |

Tableau 5.9 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour les méthodes GloVe sur le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrains la similarité issue du jeu de données MIREX

| Méthodes | précision | rappel | nDCG |
|---|---|---|---|
| TF-IDF | 0.133 | 0.093 | 0.185 |
| LSI | 0.165 | 0.112 | 0.224 |
| LDA | 0.089 | 0.059 | 0.125 |
| OV TFIDF | 0.201 | **0.144** | 0.270 |
| OV LSI | **0.202** | 0.133 | **0.274** |
| OV LDA | 0.087 | 0.061 | 0.127 |

Tableau 5.10 – Moyenne des résultats pour $k = 5$, pour le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrain la similarité issue du jeu de données MIREX.

| Méthodes | précision | rappel | nDCG |
|---|---|---|---|
| TF-IDF | 0.114 | 0.146 | 0.166 |
| LSI | 0.147 | 0.186 | 0.209 |
| LDA | 0.082 | 0.099 | 0.118 |
| OV TFIDF | 0.175 | **0.226** | 0.248 |
| OV LSI | **0.179** | 0.219 | **0.254** |
| OV LDA | 0.079 | 0.102 | 0.117 |

Tableau 5.11 – Moyenne des résultats pour $k = 10$, pour le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrain la similarité issue du jeu de données MIREX.

par la quantité de *tags* disponible pour chaque artiste. Nous avons récupéré sur *Last.fm* $\approx 25$ tags/artistes. L'association des *tags* aux autres descripteurs permet d'améliorer les résultats obtenus par ces descripteurs seuls. Les méthodes *OV* permettent d'observer que l'accumulation de plusieurs descripteurs pour former un seul vecteur permet d'obtenir en moyenne de meilleurs résultats. Mais ces méthodes sont moins efficaces que celles utilisant uniquement les *tags*.

— resultats de TF-IDF(tags) > OVTF-IDF(bios et genres et tags)

— resultats de LSI(tags) > OV LSI(bios et genres et tags)

— resultats de LDA(tags) > OV LDA(bios et genres et tags)

Ces résultats nous font aussi remarquer que l'utilisation des biographies pour caractériser les artistes n'est pas toujours efficace. Nous n'avons pas non plus cherché à faire un traitement poussé des biographies. Nous avons simplement enlevé les mots-vides avant d'appliquer la mesure de

pondération *TF-IDF*, et les méthodes de réduction de dimensionnalité (*LSI, LDA*). Nous n'avons pas cherché à extraire les concepts présents dans les biographies comme dans Oramas *et al.* [2015]. Nous remarquons même que nous obtenons dans nos expérimentations de meilleurs résultats que ceux présentés par Oramas *et al.* [2015] rien qu'en ajoutant les tags et les genres aux biographies et en faisant une réduction de dimensionnalité type *LSI*. Oramas *et al.* [2015] utilisent principalement les biographies afin de rechercher les entités nommées et les représentent sous forme de graphe. Dans nos résultats, l'application des méthodes *word2vec* et *GloVe* permet d'améliorer les résultats obtenus par l'utilisation seule de la biographie. En entrainant nos vecteurs de termes avec *word2vec*, nous avons remarqué que les vecteurs de dimension plus réduite donnaient de meilleurs résultats. Notre corpus d'apprentissage est composé des biographies des artistes et des descriptions des genres musicaux obtenus sur *DBPedia*. Pour la méthode GloVe, nous observons que les vecteurs de plus grandes dimensions donnent de meilleurs résultats.

Nous pouvons tirer quelques informations importantes de ces expérimentations. L'utilisation des *tags*, qui résultent en fin de compte de l'intelligence collective des usagers de *Last.fm*, permet obtenir de meilleurs résultats que ceux obtenus par les données récupérées sur *DBpedia*. Les tags décrivent les artistes avec des concepts qui parlent à une foule et qui permettent de mieux capturer les similarités entre les artistes. Le processus de description des artistes est peu couteux pour les administrateurs de *Last.fm*. L'utilisation de *tags* traités additionnée à des données d'écoute devrait améliorer les résultats des mesures de similarité. Nous nous rendons aussi compte au vu des résultats obtenus que les genres sont des descripteurs qui permettent de capturer aussi la similarité entre des artistes. Les genres, moins nombreux que les tags, arrivent souvent en deuxième position derrière les tags dans nos tableaux de résultats. L'utilisation de techniques de réduction de dimensionnalité pour les genres est très efficace et permet d'améliorer les résultats. Les méthodes « + » pour l'addition des distances de plusieurs descripteurs pourraient être améliorées en apprenant les paramètres de la formule 5.6 à l'aide d'une régression linéaire. Nous avons fixé ces poids à 1 et 0. La dernière notion que nous pouvons tirer de cette expérimentation est que les méthodes purement basées sur le texte sont toujours efficaces si nous comparons nos résultats à ceux obtenus par Oramas *et al.* [2015]. Cette première expérimentation nous permettent de savoir quelles données choisir et quel traitement nous devrons appliquer sur nos données en vue de mettre en place un système de recommandation basé sur le contenu [Lhérisson *et al.*, 2017c].

## 5.3 Une mesure de similarité en cas de pénurie de données

Dans cette section, nous allons présenter une méthode de calcul de dissimilarité entre des objets dont la distance entre leurs caractéristiques est connue. Cette méthode de calcul a pour but d'estimer un indice de dissimilarité/similarité entre des objets faiblement décrits. Nous avons vu dans la section précédente 5.2 que la richesse des descripteurs fait la fiabilité d'une mesure de similarité (cf. les *tags* de *Last.fm*). Or dans le cas d'un site de streaming comme *1D touch* qui ne dispose pas encore d'une grande communauté (78 248 utilisateurs inscrits pour *1D touch* contre 30 millions pour *Last.fm*) la création d'une folksonomie semble compromise. De plus, les artistes indépendants de la plateforme *1D touch* sont généralement peu ou mal décrits. Ils sont décrits seulement par des *genres* musicaux. Nous avons recensé une moyenne de deux (2) *genres* musicaux par artistes issues d'un corpus de 22 *genres* musicaux (nous sommes loin des 215 genres musicaux de l'expérimentation précédente).

Dans la section 5.2 nous avons avancé que plus des objets ont des caractéristiques semblables plus la distance entre ces objets sera petite. À contrario, plus les objets ont des caractéristiques distinctes, plus la distance sera grande. Le vecteur $A = [a, b, c]$ sera plus proche du vecteur $B = [b, c]$ que du vecteur $C = [i, c]$. La distance entre deux vecteurs $I = [a, b]$, $J = [c, d]$ dont toutes les caractéristiques sont différentes sera maximale. Ce calcul ne prend pas en compte la nuance qu'il peut exister entre les caractéristiques des objets. Deux artistes $A$ et $B$ jouant pour $A = [Rock]$ et $B = [Grunge]$ auront une distance maximale ($Dist = 1.0$ pour $Dist \in [0, 1]$). Or le grunge est un sous-style du Rock [13]. Le calcul de distance devrait prendre en compte cette information. La proximité entre les genres devrait être utilisée dans ce cas. La proximité entre les *genres* musicaux peut être estimée en utilisant une taxonomie, une ontologie et s'il n'existe pas de classification pour les *genres* musicaux, cette proximité pourrait être estimée en se basant sur la syntaxe d'écriture des genres.

### 5.3.1 Présentation de notre méthode

Nous avons proposé une méthode qui permet de calculer la distance entre deux ensembles en utilisant la distance qui existe entre les caractéristiques des ensembles. Pour illustrer notre méthode, nous avons utilisé des points dans un espace euclidien (figure 5.6). Les points dans la figure 5.6 sont dans un espace à 2 dimensions. Les points dont la distance est faible seront proches sur la figure 5.6. Notez que les données que nous aurons à traiter ne seront pas caractérisées par des données dans un espace à deux dimensions.

Les points $a, b, c, e, f, g, h, i$ et $j$ représentent les caractéristiques des objets. Nous avons utilisé la distance euclidienne pour estimer la différence entre les points du plan (tableau 5.12). À partir de ces points, nous pouvons former des ensembles, par exemple l'ensemble $A = \{a, b, c\}$. Dans nos exemples, un ensemble représente un groupe de musique ou un artiste, et les éléments

13. `http://fr.dbpedia.org/page/Grunge`

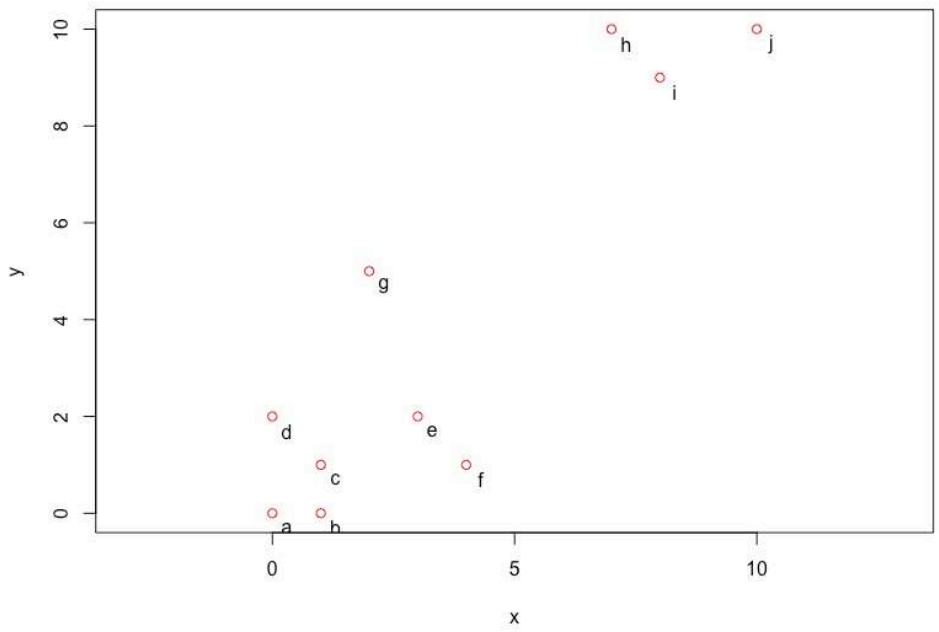



FIGURE 5.6 – Points dans un espace vectoriel.

appartenant à l'ensemble représentent les genres musicaux, ou des tags associés au groupe musical ou à l'artiste.

Pour estimer la différence entre deux ensembles, deux artistes ou groupes de musique, nous définissons une distance $\delta$ qui devra respecter les propriétés de distance :

1. La symétrie : $\forall(a,b) \in E^2,\ \delta(a,b) = \delta(b,a)$
2. La séparation : $\forall(a,b) \in E^2,\ \delta(a,b) = 0 \Leftrightarrow a = b$
3. L'inégalité triangulaire : $\forall(a,b,c) \in E^3,\ \delta(a,c) \leq \delta(a,b) + d(b,c)$

Nous voulons aussi que la distance $\delta$ entre deux ensembles soit petite : si les deux ensembles ont beaucoup d'éléments en communs ou si la distance entre les éléments est petite. Notre distance doit prendre en compte les éléments dans les ensembles et la distance qui existe entre les éléments. À titre d'exemple, si nous considérons les ensembles suivants : $A = \{a, b, c\}$, $B = \{a, b\}$, $C = \{a, b, j\}$, nous voulons que notre distance respecte la propriété suivante : $\delta(A, B) < \delta(B, C)$. En effet, $B$ est inclus dans $A$, $B$ est inclus dans $C$, et l'intersection entre $A$ et $C$ est égale à $\frac{2}{3}$, sauf que sur la figure 5.6, nous remarquons que l'élément $j$ se trouve à une distance plus éloignée des éléments $a$ et $b$ que l'élément $c$.

Le calcul de la distance entre les ensembles peut être obtenu en faisant la moyenne des distances entre les caractéristiques des ensembles, ou en prenant le maximum ou le minimum des distances entre les caractéristiques. Or la moyenne des distances entre les éléments des deux ensembles ne respecte pas la propriété de séparation entre les deux ensembles. Le choix de la distance maximale ou minimale entre les caractéristiques pour représenter la distance entre deux

| | a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|---|
| a | 0.000 | 0.070 | 0.100 | 0.141 | 0.254 | 0.291 | 0.380 | 0.863 | 0.841 | 1.00 |
| b | | 0.000 | 0.070 | 0.158 | 0.200 | 0.223 | 0.360 | 0.824 | 0.806 | 0.951 |
| c | | | 0.000 | 0.100 | 0.158 | 0.212 | 0.291 | 0.764 | 0.751 | 0.900 |
| d | | | | 0.000 | 0.212 | 0.291 | 0.254 | 0.751 | 0.751 | 0.905 |
| e | | | | | 0.000 | 0.100 | 0.223 | 0.632 | 0.608 | 0.751 |
| f | | | | | | 0.000 | 0.316 | 0.670 | 0.632 | 0.764 |
| g | | | | | | | 0.000 | 0.500 | 0.509 | 0.667 |
| h | | | | | | | | 0.000 | 0.100 | 0.212 |
| i | | | | | | | | | 0.000 | 0.158 |
| j | | | | | | | | | | 0.000 |

Tableau 5.12 – Distance euclidienne entre les points. Partie triangulaire supérieure de la matrice de distance

ensembles ne respecte pas non plus cette propriété. Par exemple, si nous reprenons l'ensemble $I = \{a, b, c, d, e, f, g, h, i\}$, la moyenne des distances de $(Dist(I, I))$ est 0.400, le maximum est 0.863 et le minimum est 0.070. Donc, nous nous sommes inspirés de la distance de Jaccard, $J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$ pour proposer un calcul de distance entre les éléments des ensembles. La distance de Jaccard est utile pour étudier la dissimilarité entre des objets constitués d'attributs binaires. Dans notre cas, il s'agit de la présence ou non d'un genre parmi les caractéristiques d'un artiste. Dans la distance de Jaccard, plus l'intersection entre deux ensembles est grande, plus les ensembles auront une distance faible. Si les deux ensembles sont définis par les mêmes individus, la valeur de la distance de Jaccard entre ces deux ensembles sera égale à 0. Si nous reprenons l'ensemble $I$, la distance de Jaccard(I,I) est égale à 0 $(1 - \frac{9}{9})$.

Mais la formule de la distance de Jaccard ne prend pas en compte les éléments qui sont inclus dans un ensemble et qui ne sont pas inclus dans l'autre. Nous avons ajouté à la formule de la distance de Jaccard la distance entre ces éléments. Pour tout couple d'ensembles $A$, $B$ nous représentons par $M_{01}$ les éléments $\in B$ et $\notin A$ et par $M_{10}$ les éléments $\in$ dans $A$ et $\notin B$. Ainsi, cette distance, qui respecte la propriété de séparation, est calculée suivant la formule 5.9 :

$$\delta(A, B) = \frac{(\text{Jaccard}(A, B) + \text{moyenne\_distance}(M_{10}, M_{01}))}{2}; \tag{5.9}$$

En utilisant cette formule sur la distance entre les ensembles $A = \{a, b, c\}$, $C = \{a, b, j\}$, nous remarquons que la présence des points $c$, $j$ augmente la distance entre les ensembles, et la présence des points $a$, $b$ communs aux deux ensembles les rapprochent. En appliquant la formule 5.9 pour calculer la distance entre ensembles $A$ et $C$ nous obtenons une distance de Jaccard de 0.5, nous obtenons 0.9 pour la moyenne des distances entre $M_{10}$, $M_{01}$ et $\delta(A, C) = 0.7$. En revanche, si nous testons cette formule sur les ensembles $A$, $B$ $(\delta(A, B))$ et les ensembles $B$, $C$ $(\delta(B, C))$ nous obtenons les mêmes résultats. Nous obtenons $Jaccard(A, B)$ et $Jaccard(B, C) = 0.333$, et 0 pour la moyenne des distances entre $M_{10}$, $M_{01}$. Finalement nous obtenons, $\delta(A, B) = \delta(B, C) = 0.166$.

Or nous voulions que notre distance prenne en compte la présence des éléments distants dans les ensembles. Dans notre exemple, l'élément $j$ devrait pénaliser $\delta(B, C)$ et on devrait obtenir $\delta(A, B) < \delta(B, C)$. Dans cet exemple, comme nous l'avons dit plus haut, nous avons $B \subset A$ et $B \subset C$. Donc pour arriver trouver la différence entre les ensembles, nous avons pris en compte la distance entre tous les points qui sont propres à un ensemble et ceux qui sont dans l'intersection (formules 5.10 et 5.11).

$$\text{distance\_Inter\_c}_1 = \text{distance}(M_{10}, \text{intersection}(A, B)); \tag{5.10}$$

$$\text{distance\_Inter\_c}_2 = \text{distance}(M_{01}, \text{intersection}(A, B)); \tag{5.11}$$

Pour obtenir la distance entre les intersections, nous avons posé 4 conditions. Elles dépendent de la valeur des distances $distance\_Inter\_c_1$ et $distance\_Inter\_c_2$. Si les deux valeurs de distance sont nulles, la distance entre les intersections est nulle. Si l'une des distances est nulle, la distance entre les intersections prend la valeur de la distance non nulle. Si les deux distances sont supérieures à zéro, la distance entre les intersections est égale à la moyenne des deux distances.

La distance $\delta$ entre deux ensembles est ainsi modifiée comme indiqué en formule 5.12. Cette distance a la particularité de respecter les propriétés de distance et les conditions que nous avions définies préalablement.

$$\delta(A, B) = \frac{Jaccard(A, B) + \left(\frac{\text{moyenne\_distance}(M_{10}, M_{01}) + \text{distance\_Inter\_c}_1\text{\_c}_2}{2}\right)}{2} \tag{5.12}$$

## 5.3.2 Expérimentations et résultats

### 5.3.2.1 Jeux de données

Nous avons réalisé des expériences sur le jeu de données de 268 artistes fournis par Oramas *et al.* [2015]. Nous avons étendu ce corpus en récupérant 215 genres musicaux sur *DBPedia* ainsi que leur texte descriptif. Nous n'avons pas utilisé d'autres descripteurs. Ce corpus a, pour chaque artiste, au plus 10 artistes similaires. Ces artistes similaires nous servent de vérité terrain dans l'évaluation des méthodes développées.

### 5.3.2.2 Les méthodes évaluées

Nous avons évalué la distance $\delta$ définie dans la formule 5.12 et calculée suivant l'algorithme 1. Cette distance $\delta$ nécessite en entrée (sur les données que nous utilisons) la distance entre les genres musicaux. Pour estimer cette distance, nous avons utilisé deux sources de données. Nous avons utilisé la cooccurrence des genres dans les listes de genres joués par les artistes (méthode *CooD*). Nous avons utilisé les descriptions des genres issues de *DBPedia* (méthode *DBD*). Nous réalisons ensuite une réduction de dimensionnalité sur les matrices obtenues à partir de ces deux

**Algorithme 1 :** calcul de distance pour les données faiblement décrites

**Data** :
$D$ : Matrice de distance entre les descripteurs,
$U$ : Ensemble composé de descripteurs,
$V$ : Ensemble composé de descripteur
**Result** : distance entre les deux ensembles $\delta$
**begin**
  $intersection = U \cap V$
  $c_1 = U$
  $c_2 = V$
  **if** $intersection$ **then**
    $c_1 = \complement_{intersection} U$
    $c_2 = \complement_{intersection} V$
  $distance\_Inter\_c_1 = \text{distance}(c_1, intersection)$
  $distance\_Inter\_c_2 = \text{distance}(c_2, intersection)$
  **if** $distance\_Inter\_c_1 > 0$ *and* $distance\_Inter\_c_2 > 0$ **then**
    $distance\_Inter\_c_1\_c_2 = (distance\_Inter\_c_1 + distance\_Inter\_c_2)/2$
  **if** $distance\_Inter\_c_1 > 0$ *and* $distance\_Inter\_c_2 == 0$ **then**
    $distance\_Inter\_c_1\_c_2 = distance\_Inter\_c_1$
  **if** $distance\_Inter\_c_1 == 0$ *and* $distance\_Inter\_c_2 > 0$ **then**
    $distance\_Inter\_c_1\_c_2 = distance\_Inter\_c_2$
  **if** $distance\_Inter\_c_1 == 0$ *and* $distance\_Inter\_c_2 == 0$ **then**
    $distance\_Inter\_c_1\_c_2 = 0$
  $\delta = \frac{Jaccard(U,V) + \left(\frac{moyenne\_distance(c_1,c_2) + \text{distance\_Inter\_c}_1\text{\_c}_2}{2}\right)}{2}$

sources de données (vecteurs de rang = 10). Nous avons volontairement limité le nombre de genres par artistes à 1 et à 2. Notre voulions voir si notre méthode est une bonne alternative aux méthodes traditionnelles (distance de Jaccard) dans les situations où les données sont très peu décrites.

#### 5.3.2.3 Résultats et discussion

Pour évaluer et comparer les différentes méthodes, nous avons utilisé les mesures de performances standards, la Précision@k, le Rappel@k et la mesures nDCG@k pour $k = 5$, $k = 10$. Nous avons comparé nos méthodes à la distance de Jaccard. Les résultats sont présentés dans le tableau 5.13. Le *ratio genres/artistes* est le nombre de genres par artistes retenus. Nous avons pris les genres au hasard pour chaque artiste. Sur les 215 genres initiaux après la sélection aléatoire, il ne nous restait que 68 genres pour un *ratio genres/artistes* = 1 et 121 pour un *ratio genres/artistes* = 2. Nous avons mis en gras les meilleurs résultats et en italique les deuxièmes meilleurs résultats pour chaque combinaison de paramètres ($k$ et *ratio genres/artistes*). Nous remarquons que notre méthode *CooD* basée sur la cooccurrence obtient en général les meilleurs résultats. Elle arrive même à améliorer le résultat de la distance de Jaccard de 1% pour $k = 5$ et *ratio genres/artistes* = 1. En revanche notre deuxième méthode *DBD* ne réussit par à battre la distance de Jaccard dans toutes les configurations. Nous utilisons pour cette méthode la description *DBPedia* des genres musicaux. Or, nous avions vu dans l'expérimentation précédente (section 5.2.3) que l'usage des biographies pour les artistes ne donnait pas des résultats encourageants. De plus la méthode *DBD* : perd en précision quand nous augmentons le ratio de genres par artistes. Cependant notre méthode *CooD* semble adopter le comportement inverse. La méthode de Jaccard devient elle aussi plus efficace quand le nombre de genres disponibles s'élève.

Nous pouvons tirer comme information que notre calcul de distance $\delta$ est un bon compromis à la distance de Jaccard quand les données sont très peu décrites. Cette méthode est tributaire de la distance entre les descripteurs. Nous avons vu qu'en utilisant la cooccurrence des descripteurs nous avons pu obtenir une bonne distance $\delta$. Conséquemment, nous faisons l'hypothèse qu'avec une bonne catégorisation des descripteurs (p.ex., un thésaurus) nous pouvons améliorer la précision de la distance $\delta$.

## 5.4 Conclusion

La similarité/dissimilarité est une notion importante dans les systèmes de recommandation. Dans ce chapitre, nous avons étudié cette notion en cherchant à retrouver des similarités entre les articles (des artistes). La précision de cet indice de similarité dépend grandement des descripteurs des articles. Des descripteurs fins produiront de très bons indices de similarité. Ces mêmes descripteurs seront utilisés aussi dans un système de recommandation sur le contenu pour qualifier les articles, mais aussi les utilisateurs. Les descripteurs peuvent être présents dans

| Méthodes | k | ratio genres/artistes | précision | rappel | nDCG |
|---|---|---|---|---|---|
| Jaccard | 5 | 1 | 0.025 | 0.016 | 0.037 |
| CooD | 5 | 1 | **0.035** | **0.024** | **0.058** |
| DBD | 5 | 1 | *0.031* | *0.021* | *0.053* |
| Jaccard | 5 | 2 | 0.031 | 0.021 | 0.048 |
| CooD | 5 | 2 | **0.033** | **0.022** | **0.047** |
| DBD | 5 | 2 | 0.030 | 0.020 | 0.046 |
| Jaccard | 10 | 1 | *0.025* | *0.031* | 0.037 |
| CooD | 10 | 1 | **0.030** | **0.037** | **0.050** |
| DBD | 10 | 1 | 0.023 | 0.029 | *0.042* |
| Jaccard | 10 | 2 | *0.031* | *0.039* | *0.046* |
| CooD | 10 | 2 | **0.037** | **0.048** | **0.051** |
| DBD | 10 | 2 | 0.030 | 0.038 | 0.045 |

Tableau 5.13 – Précision, rappel et nDCG à $k = 5$ et $k = 10$, pour le jeu de données de 268 artistes tirés de *Last.fm* avec pour vérité terrain la similarité issue du jeu de données MIREX dans le cas où les données sont peu décrites

le jeu de données. Mais nous avons montré qu'il peut être judicieux d'aller rechercher les descripteurs sur le *web*, sur des sites spécialisés ou sur des encyclopédies générales. Nous avons vu que l'utilisation de plusieurs descripteurs peut permettre d'obtenir une bonne estimation de la similarité entre les articles. Nous avons aussi vu que certains descripteurs peuvent permettre d'être encore plus précis. Mais parfois, les articles sont très peu qualifiés et non indexés sur le *web*. Nous avons démontré que dans cette situation (qui est aussi celle des données d'*1D touch*) l'application des métriques de similarité et de dissimilarité habituelles ne permet pas d'avoir une estimation précise de la proximité qui peut exister entre les articles. Nous avons présenté une métrique de distance qui permet de répondre à cette problèmatique. Nous proposons dans le chapitre suivant des modèles de recommandation qui utilisent les prétraitements ainsi que les calculs de distance présentés dans ce chapitre.

Chapitre 6

# Comment des recommandations peuvent-elles être audacieuses ?

Dans ce chapitre, nous allons présenter les différents modèles de recommandation que nous avons développés. Nous avons cherché à produire des recommandations audacieuses. Nous définissons une liste de recommandations audacieuses comme un ensemble de produits qui sont différents entre eux et sensiblement différents des produits considérés dans le profil de l'utilisateur. Nos recommandations audacieuses doivent dépasser les limites que les utilisateurs s'imposent et cherchent à les emmener, sans les perdre, dans des chemins de découvertes inédits. Nos algorithmes doivent avoir l'audace de proposer aux utilisateurs des contenus qui sortent de leurs habitudes de consommation. Nos recommandations audacieuses doivent aussi avoir la vertu d'assurer une couverture maximale du catalogue, c'est-à-dire ne pas favoriser les articles les plus populaires.

Ces recommandations dans le domaine culturel permettront d'étendre les connaissances culturelles d'un public en fournissant des contenus inédits, qui ne seront pas nécessairement les grands noms habituels connus de tout le monde. Nos recommandations audacieuses sont un ensemble de techniques de recommandation qui aboutissent à des propositions de consommation de biens culturels différents, mais pas trop éloignés, proches, mais pas trop similaires au regard d'un état de consommation culturelle donné d'une personne.

Dans la suite de ce chapitre, nous allons présenter un premier modèle que nous avons nommé *Maximal Personalized Diversification Method* (méthode de diversification personnalisée optimale) (section 6.1). Ce modèle se base sur des techniques de *clustering* pour connaitre les différents centres d'intérêt d'un utilisateur et produire des recommandations qui restent différentes entre elles, mais proches des différents centres d'intérêt identifiés de l'utilisateur. Nous présentons par la suite un deuxième modèle que nous avons nommé *Mexican-Hat Diversity Model* (modèle basé sur le Chapeau mexicain)(section 6.2) qui utilise une ondelette pour modéliser les zones de découvertes pour chaque utilisateur.

## 6.1 MPD : Recommandations audacieuses basées sur le partitionnement

Les systèmes de recommandation jouent un rôle principal dans la prédiction des interactions futures des utilisateurs dans un système. Les avancées qui ont été faites dans le traitement du langage naturel, tel que le plongement de mots, offrent des outils utiles qui créent un regain d'intérêt pour les systèmes basés sur le contenu. Cette méthode couplée aux méthodes existantes d'analyses des thèmes latents permet de mieux définir les profils des utilisateurs et prédire les articles que les utilisateurs aimeraient consommer à l'avenir en fonction d'une mesure de similarité (chapitre 5). Cependant, le défi reste de savoir comment équilibrer la similarité et la diversité (chapitre 2). La recherche de la précision dans les systèmes basés sur le contenu est faite en sélectionnant les articles les plus similaires à ce que l'utilisateur avait aimé dans le passé. Cela produit un manque de diversité et induit de la monotonie et de la frustration chez les utilisateurs (chapitre 2). Or l'être humain est friand de nouveauté et souhaite qu'on le sorte de sa zone de confort. Les concepts de nouveauté et diversité abordés dans les systèmes de recommandation vont dans ce sens en apportant une solution face à la personnalisation accrue et à l'uniformisation des goûts (chapitre 3).

Nous nous sommes intéressés à trouver une liste de $n$ articles pour chaque utilisateur qui sont pertinents, mais aussi divers. Nous avons cherché à trouver un compromis entre diversité et pertinence. Il a été montré que les algorithmes de recommandation basés sur le filtrage collaboratif étaient plus adaptés pour amener de la diversité dans les listes de recommandations [Channamsetty et Ekstrand, 2017]. Dans la suite de cette section, nous allons présenter deux nouvelles méthodes pour les systèmes de recommandation basés sur le contenu qui favorisent la personnalisation et la diversité. Pour ce faire, nous avons combiné la méthode *Max-Sum Diversification* [Borodin *et al.*, 2012] et des techniques de *clustering*. Dans le reste de cette section, nous allons décrire notre processus de modélisation dans un espace vectoriel pour les utilisateurs et les articles (section 6.1.1), et nous allons par la suite présenter nos deux modèles de recommandations qui combinent la pertinence et la diversité (section 6.1.2).

### 6.1.1 Modèlisation dans un espace vectoriel

#### 6.1.1.1 Formulation du problème et notation

Nous considérons que nous avons un ensemble fini d'utilisateurs $U = \{u_1, ..., u_m\}$, et un ensemble fini d'articles $I = \{i_1, ..., i_n\}$.

Pour chaque article, nous avons une description textuelle de cet article $Desc = (desc_1, ..., desc_n)$, et une liste de *tags* tiré de l'ensemble $T = \{t_1, ..., t_l\}$. Les articles peuvent aussi être décrits par une liste de genres tirés de $G = \{g_1, ..., g_o\}$.

Nous voulons produire un ensemble de listes de recommandation, $S = \{s_1, ..., s_m\}$, avec pour chaque utilisateur une liste de longueur $k$ représentant les articles qui lui sont recommandés

en nous basant sur la consommation passée ou les appréciations passées de l'utilisateur $P = \{p_1, ..., p_m\}$.

Notre objectif final est d'avoir pour chaque $s \in S$, une liste personnalisée de $k$ articles qui est la plus diverse et la plus pertinente.

#### 6.1.1.2 Modélisation des articles

Pour produire une représentation pour les articles, nous avons utilisé des techniques de l'état de l'art pour analyser les textes descriptifs des articles (section 5.2). Nous construisons une matrice en comptant la fréquence de chaque mot du vocabulaire dans le texte. Après la construction de cette matrice, elle est transformée en une matrice *tf-idf* (*term-frequency inverse-document-frequency*).

La matrice *tf-idf* est par la suite réduite en utilisant l'indexation sémantique latente (*LSI*)(section 5.2.2.1). Cette méthode est utilisée pour décomposer la matrice en un nombre de facteurs orthogonaux, qui sont ensuite utilisés pour représenter les descriptions des articles en tant que vecteurs. Nous avons fait le choix de *LSI* puisque cette méthode donnait de meilleurs résultats dans nos expérimentations (section 5.2.3).

Nous avons aussi créé des vecteurs latents à partir des matrices articles-tags et articles-genres, en utilisant la même méthode décrite plus haut. Ces vecteurs de tags et de genres sont utilisés pour améliorer la représentation des articles. Nous écrivons $\mathbf{i_n}$ le vecteur représentant le énième article.

#### 6.1.1.3 Modélisation des utilisateurs

Une fois que les vecteurs des articles sont calculés, une représentation pour chaque utilisateur peut être calculée en utilisant les vecteurs de représentation des articles et les données d'écoute de l'utilisateur. Nous avons défini deux façons d'obtenir une représentation pour l'utilisateur.

**6.1.1.3.1 Modélisation des utilisateurs par la somme des vecteurs des articles** Pour cette représentation, nous faisons la somme des vecteurs des articles avec lesquels un utilisateur a interagi. Nous normalisons le vecteur, et cela donne un vecteur de préférence global pour chaque utilisateur (formule 6.1).

$$u = \sum i \in P_u \tag{6.1}$$

**6.1.1.3.2 Modélisation du profil de l'utilisateur suite à une factorisation de matrice** Nous avons aussi utilisé des techniques de factorisation de matrice pour représenter le profil d'un utilisateur en lieu et place d'une modélisation par somme des vecteurs des articles. Étant donné que les interactions utilisateurs-articles contiennent généralement des évaluations quantitatives ou le nombre de fois que l'utilisateur interagit avec l'article, nous nous sommes

inspirés des méthodes de filtrage collaboratif pour trouver un vecteur utilisateur [Hu *et al.*, 2008; Koren *et al.*, 2009; Takács et Tikk, 2012]. Nous utilisons les vecteurs des articles (section 6.1.1.2) et à partir de ces vecteurs, nous apprenons les facteurs pour les utilisateurs. Nous utilisons la méthode simple, *SVD* de Koren [2008], lorsque nous avons des retours explicites des utilisateurs sous forme d'évaluations, et la méthode *Alternating Least Squares method* de Koren *et al.* [2009] quand nous avons des retours implicites d'utilisateurs (par exemple le nombre d'écoutes pour un morceau de musique).

Nous avons implémenté ces approches en décomposant la matrice utilisateur/article en des facteurs utilisateurs ($p$), articles ($q$) de plus petites dimensions. Nous estimons les préférences de l'utilisateur en multipliant les facteurs utilisateurs aux facteurs articles (Formule 6.2). Pour apprendre ces facteurs, nous avons minimisé la fonction quadratique (Formule 6.3).

Pour l'approche *SVD* de Koren [2008], pour chaque exemple de la base de données, il faut calculer l'erreur $(r_{ui} - p_u^T q_i)^2$ puis mettre à jour les paramètres en utilisant la descente de gradient stochastique et en allant dans la direction opposée du gradient. La méthode *Alternating Least Squares method* [Koren *et al.*, 2009] est différente de la méthode *SVD* [Koren, 2008]. Comme son nom peut l'indiquer, dans cette méthode l'algorithme estime en premier $p$ en utilisant $q$ puis estime $q$ en utilisant $p$. Après un nombre d'itérations déterminé, l'algorithme atteint un point de convergence quand les facteurs $p$ et $q$ ne changent pas ou que les changements sont minimes.

$$r'_{ui} = p_u^T q_i \tag{6.2}$$

$$argmin_p \sum_{u,i} (r_{ui} - p_u^T q_i)^2 \tag{6.3}$$

Pour les deux méthodes, nous fixons les vecteurs des articles au début du processus d'apprentissage. Les vecteurs des utilisateurs appris sont dans le même espace vectoriel que les vecteurs des articles (section 1.2.1.2).

Dans le reste du manuscrit, nous appellerons cette méthode SVD-MPD.

#### 6.1.1.4 Modélisation des utilisateurs pour un apport en diversité

Nous présentons ici la modélisation des préférences des utilisateurs qu'utilisent les méthodes que nous allons présenter dans la suite (section 6.1.2).

**6.1.1.4.1 Modélisation des utilisateurs par un clustering** Nous avons utilisé la méthode de partitionnement en k-moyennes (*k-means*) pour partitionner les vecteurs des articles associés à un utilisateur en $k$ groupes. Chaque groupement (*cluster*) représente un type d'articles que l'utilisateur apprécie. Pour produire un bon groupement, nous avons cherché le nombre $k$ optimal de groupements. Ce paramètre $k$ dépend de la cohésion des clusters et de la séparation des clusters. La cohésion d'un cluster est la somme des poids de tous liens qu'il y a dans un cluster.

Dans la pratique, cette valeur est égale à la somme des carrés des distances dans le cluster (*WSS : within-cluster sum of squares*). La séparation des clusters est la somme des poids entre les noeuds du cluster et les noeuds à l'extérieur du cluster. En pratique, cette valeur peut-être calculée par la somme des carrés des distances entre les clusters (*BSS : between-cluster sum of squares*).

Un partitionnement sera considéré meilleur qu'un autre si cela minimise l'inertie intra-cluster (par exemple, *WWS* qui représente l'inertie intra cluster) et maximise l'inertie inter-cluster (par exemple, *BBS* qui représente l'inertie inter cluster). Nous avons utilisé l'index de *Calinski-Harabasz* (CH, formule 6.4) [Calinski et Harabasz, 1974] pour obtenir le nombre de clusters ayant à priori de bonnes propriétés pour un utilisateur.

$$CH = \frac{BSS/(k-1)}{WSS/(n-k)} \tag{6.4}$$

Une fois que les éléments associés à un utilisateur ont été regroupés en un ensemble noté $C$ de clusters, nous utilisons les coordonnées moyennes de chaque cluster $c$ pour modéliser une préférence ou un genre spécifique que l'utilisateur aime. Dans le cas où un utilisateur n'a pas interagi avec plus de deux (2) articles, nous utilisons les vecteurs de ces articles comme des coordonnées de clusters. Nous obtenons un ensemble de vecteurs pour un utilisateur (formule 6.5).

$$\mathbf{u} = \{c_1, c_2..., c_{|C|}\} \tag{6.5}$$

## 6.1.2 Méthodes proposées

Nous expliquons les systèmes de recommandation que nous avons proposés. Nous expliquons une première méthode basée sur la similarité : la méthode *Clustering des préférences utilisateurs*. Nous détaillons ensuite un algorithme glouton qui fournit une diversification personnalisée pour l'utilisateur. Nous l'appelons *Maximal Personalized Diversification*.

### 6.1.2.1 Méthode basée sur la similarité : Clustering des préférences des utilisateurs

Le système trouve pour un utilisateur parmi les articles qu'il n'a pas consommés, les articles les plus proches de chacun des clusters qui représentent son profil. La liste finale des recommandations est produite en itérant à travers chaque cluster et en prenant l'élément le plus similaire. Des approches semblables de *clustering* de préférence d'utilisateur ont été utilisées dans le passé [Zhang et Hurley, 2008]. Mais notre approche diffère de celle présentée par Zhang et Hurley [2008] en ce sens que nous traitons chaque cluster de manière égale, en itérant sur chaque cluster, et en prennant le prochain élément le plus semblable à chaque cluster jusqu'à ce que nous ayons le nombre attendu de recommandations. Cette approche se base sur le clustering pour apporter de la diversité dans la liste de recommandation d'un utilisateur. La diversité est comprise ici comme une liste d'objets ayant des caractéristiques différentes [Castells *et al.*, 2015]. Le clustering nous permet d'identifier les sous-préférences d'un utilisateur de manière implicite. Cette méthode

nous permet de faire l'hypothèse que plus un utilisateur aura montré de centres d'intérêt, plus sa liste de recommandations sera diversifiée.

Notre approche initiale se base principalement sur le contenu. Nous construisons un espace vectoriel pour représenter les articles en fonction des caractéristiques des éléments (descriptions textuelles, genres, *tags*). Dans cette approche, nous ne nous servons pas que des vecteurs articles qui résultent de la réduction de dimensionnalité de la matrice d'utilité comme Zhang et Hurley [2008]. Notre approche prend toutes les interactions des utilisateurs pour construire son profil. Nous ne nous basons pas seulement sur les dernières interactions d'un utilisateur comme pour Vargas et Castells [2013]. La construction de la liste diversifiée à partir de cette méthode est donnée par l'algorithme 2.

**Algorithme 2 :** Clustering des préférences des utilisateurs

**Data** : $P$ : ensemble des articles appréciés par un utilisateur, $C$ : ensemble des clusters pour un utilisateur, $k$ : nombre de recommandations
**Result** : $S$ : ensembles des articles diverses
**begin**
  $S = \{\}$
  **for** $c$ *in* $C$ **do**
    **if** $|S| < k$ **then**
      $i^* = argmax_{i \notin P \wedge S} cosinus(i, c)$
      $S = S \cup \{i^*\}$

#### 6.1.2.2 Méthode de clustering personnalisée : *Maximal Personalized Diversification*

Nous introduisons ici un algorithme qui cherchera à produire une liste de recommandations diversifiée pour un utilisateur (algorithme 3). L'algorithme que nous présentons est basé sur la méthode appelée *Max-Sum Diversification* [Borodin *et al.*, 2012] qui est une combinaison linéaire d'une fonction qui mesure la pertinence d'un article ($i$) pour un utilisateur et une fonction qui mesure la différence entre les articles sélectionnés (formule 6.6).

$$\underset{S \subseteq I \setminus I_u}{\operatorname{argmax}} \left( \lambda r(i) - (1 - \lambda) \sum_{i \in S} \sum_{j \in S - i} dist(i, j) \right) \; tq \;\; |S| \leq k \tag{6.6}$$

Cette équation permet de sélectionner une liste d'articles de longueur $k$, pertinents, et divers selon la mesure de distance sélectionnée. Cette équation est composée d'une partie submodulaire, mais également d'une partie super-modulaire : la somme des distances. Les fonctions modulaires peuvent être caractérisées par un gain marginal décroissant quand la taille de l'ensemble à obtenir ou à rechercher augmente. Le problème *Max-sum Diversification* est un problème NP-difficile et peut être résolu par un algorithme glouton. Il a été démontré que l'usage d'un tel algorithme permettait d'obtenir une approximation constante d'un ratio de 2 quand il y a une contrainte de cardinalité [Borodin *et al.*, 2012].

Nous avons proposé un algorithme qui utilise une fonction qui s'écrit comme la fonction de *Max-sum Diversification*. Nous avons également une fonction submodulaire et une fonction super-modulaire. Notre fonction se compose d'une fonction $r(i)$ qui donne la pertinence de l'article $i$ pour un utilisateur, et une fonction $Div(i)$ qui a pour rôle d'ajouter de la diversité dans la liste d'éléments sélectionnés (Formules 6.7, 6.8).

$$MPD = \underset{i \in B \setminus S}{\operatorname{argmax}} \left(\lambda \cdot r(i) + (1 - \lambda) Div(i)\right) \tag{6.7}$$

$$Div(i) = \underset{c \in C}{\operatorname{argmax}} \left( sim(i, c) \cdot \frac{1}{|S|} \sum_{j \in S} Dist(i, j) \right) \tag{6.8}$$

La fonction $r(i)$ de pertinence correspond au score obtenu par l'article dans un algorithme de recommandation du filtrage collaboratif pris dans l'état de l'art. Nous appelons pour $r(i)$, la similarité entre le profil de l'utilisateur et le profil de l'article $i$ (section 6.1.1.3.1, création du profil par la somme des articles ou suivant une factorisation de matrice). $B$ correspond à une liste de recommandations obtenue par un algorithme de filtrage collaboratif que nous reclassons. Notre fonction $D(i)$ diffère de la fonction de diversité brute du *Max Sum Diversification*. En effet, nous avons ajouté ici le profil clusterisé de l'utilisateur, et nous cherchons à prendre pour chaque cluster $c$ de $C$ un article qui est à la fois proche du cluster $sim(i, c)$ et différent des autres éléments de la liste. Nous avons ajouté l'article avec la similarité maximale à un cluster pour nous assurer que même si l'article ajouté est différent des autres articles de la liste, il satisfait également l'une des sous-préférences de l'utilisateur représenté par le vecteur *centroid* du cluster offrant la diversité et personnalisation. Cette méthode utilise aussi un paramètre $\lambda$ qui permet de faire un compromis entre pertinence et diversité.

Cette méthode se base sur une méthode de clustering pour définir la diversité d'un profil d'utilisateur. Nous avons décidé d'utiliser une méthode basée sur le contenu des articles pour apporter de la diversité. Les algorithmes basés sur le contenu produisent des listes plus ou aussi diversifiées que le profil de l'utilisateur [Channamsetty et Ekstrand, 2017]. Notre stratégie est de développer des algorithmes de recommandation qui répondent aux caractéristiques du profil des utilisateurs. Dans la suite, nous avons proposé un algorithme qui étudie aussi le profil de l'utilisateur, et qui produit des recommandations à partir des articles singuliers du profil (section 6.2). L'algorithme ne nécessite pas de faire appel à un paramètre $\lambda$ pour faire un compromis entre la diversité et pertinence.

## 6.2 Recommandations audacieuses basées sur une ondelette

Alors que la section précédente abordait le problème de diversification, nous abordons ici le problème de la personnalisation accrue dans les systèmes de recommandation. Nous nous étions concentrés sur la modélisation du profil des utilisateurs en sélectionnant les articles les

**Algorithme 3 :** Maximum Personnalized Diversification

**Data** : $B$ : ensemble d'articles présélectionnés par une baseline, $P$ : ensemble d'éléments appréciés par un utilisateur, $k$ : nombre de recommandations
**Result** : $S$ : liste diversifiée d'éléments
**begin**
  $S = \{\}$
  **while** $|S| < k$ **do**
    $i^* = argmax_{i \in B}\, MPD(P, S \cup i)$
    $S = S \cup \{i^*\}$
    $B = B \backslash \{i*\}$

plus divers qu'ils ont aimés dans le passé. Nous avions ensuite créé une liste de $k$ éléments en sélectionnant les articles que nous voulions recommander dans une zone où nous considérions qu'un utilisateur allait trouver des articles différents de ce qu'il avait apprécié dans le passé [Muhlenbach *et al.*, 2017; Lhérisson *et al.*, 2017b].

Maintenant, pour contrer les effets de personnalisation accrue et de bulle de filtre [Pariser, 2011], il faut filtrer l'information de telle sorte que l'utilisateur soit conscient de la diversité des articles qui s'offrent à lui. Certaines expériences montrent que de nombreux articles du catalogue ne sont pas recommandés dans les listes de $k$ éléments proposés aux utilisateurs, à moins que le système de recommandation soit développé spécifiquement pour effectuer cette tâche [Seyerlehner *et al.*, 2009]. Le rôle d'un système de recommandation doit être de proposer des articles nouveaux et diversifiés aux utilisateurs (section 6.1). En effet, même avec un choix presque illimité d'articles, il a été montré que les utilisateurs des sites de *streaming* s'agglutinent autour d'objets populaires et ne passent pas de temps de navigation à chercher des articles de niche [Steck, 2011]. Cependant présenter une liste originale et diverse d'articles peut aussi avoir un impact négatif sur l'utilisateur [Ekstrand *et al.*, 2014]. Une liste composée d'éléments trop peu familiers peut induire de la frustration chez l'utilisateur et cela peut le mener à arrêter d'utiliser le service.

Les utilisateurs ont des réactions positives face à des listes de recommandation diverses [Ekstrand *et al.*, 2014]. Parallèlement, chaque article dans une liste diverse de recommandation peut être considéré comme un élément nouveau par rapport au reste de la liste [Vargas, 2015]. Tout en mettant l'accent sur la diversité, nous pouvons également améliorer la nouveauté dans la liste de recommandations. Ainsi, au lieu de proposer seulement des recommandations variées, mais dans les zones connues de l'utilisateur, nous allons chercher à offrir aux utilisateurs une liste de recommandations diverses sur laquelle s'appliquera un modèle de recommandation qui étendra le spectre des intérêts des utilisateurs. Dans le domaine des œuvres culturelles, nous faisons l'hypothèse qu'une telle recommandation contribuera à un enrichissement culturel.

La création de liste d'articles divers à proposer à un utilisateur est habituellement résolue en trouvant un compromis entre la précision (pertinence) et la diversité. Nous proposons de suivre cette approche, mais en mettant l'accent sur la diversité et la nouveauté. Nous nous plaçons dans

le cadre de recommandations de produits culturels. Nous partons du profil d'un utilisateur (c'est-à-dire les articles qu'il a consommés), nous cherchons à définir les éléments les plus singuliers de ce profil, puis à partir de cette liste d'articles, nous créons une liste de recommandations. Nous nous basons sur la diversité du profil de l'utilisateur pour amener de la diversité, mais la méthode de recommandation proposée apporte de la diversité et de la nouveauté en cherchant des articles qui ne sont ni trop similaires, ni trop différents de ce que l'utilisateur a déjà consommé. Dans le reste de cette partie, nous allons rappeler les formulations (section 6.2.1), présenter notre méthode d'identification de la diversité dans le profil d'un utilisateur (section 6.2.2), et finir par présenter la fonction de recherche de nouveauté et de diversité (section 6.2.3).

### 6.2.1 Formulation du problème et notation

Pour résoudre le problème de la personnalisation accrue dans les systèmes de recommandations, nous avons introduit dans notre approche une fonction objectif et un modèle que nous avons appelé « *Mexican-Hat Diversity Model* » (MHDM). Nous avons aussi proposé une méthode qui permet d'identifier la diversité dans le profil d'un utilisateur (c.-à-d., les articles qu'il a consommés dans le passé). Les solutions pour ces deux méthodes peuvent être trouvées en appliquant un algorithme glouton. Pour expliquer le fonctionnement de ces algorithmes, il est nécessaire de rappeler certains éléments de notations.

Nous considérons que nous avons un ensemble fini d'utilisateurs $U = \{u_1, ..., u_m\}$ et un ensemble fini d'articles $I = \{i_1, ..., i_n\}$. Les articles sont plus ou moins proches les uns des autres en fonction d'une fonction de distance *Dist* basée sur les contenus décrivant ces articles.

Nous voulons présenter chaque utilisateur $u$ de l'ensemble $U$, une liste de recommandation $S_u$ de $k$ articles en se basant sur les articles consommés et appréciés dans le passé $P_u$. Chaque utilisateur $u$ a consommé un ensemble $P_u$ d'articles avec $P_u \subset I$.

Notre objectif est de produire une liste de recommandations $S_u$ contenant un ensemble d'articles avec la diversité la plus intéressante et la nouveauté la plus intéressante pour un utilisateur.

### 6.2.2 Identification de la diversité dans le profil de l'utilisateur

La première étape consiste à identifier la diversité dans le profil d'un utilisateur en sélectionnant un nombre limité, mais très diversifié, d'éléments considérés comme les éléments sources qui vont permettre de trouver les recommandations diversifiées. Nous considérons l'ensemble $Gr_u$ contenant les éléments sources diversifiés extraits de l'ensemble $P_u$ des articles préférés par un utilisateur.

L'identification de la diversité dans le profil d'un utilisateur peut être liée au problème qui consiste à trouver l'enveloppe convexe d'un objet ou d'un regroupement d'objets en géométrie. L'enveloppe convexe d'un objet géométrique est l'ensemble convexe le plus petit parmi ceux qui

le contiennent (Figure 6.1). Pour mémoire, un ensemble est dit convexe, lorsque, chaque fois qu'on y prend deux points $A$ et $B$, le segment $[A, B]$ qui les joint y est entièrement contenu.

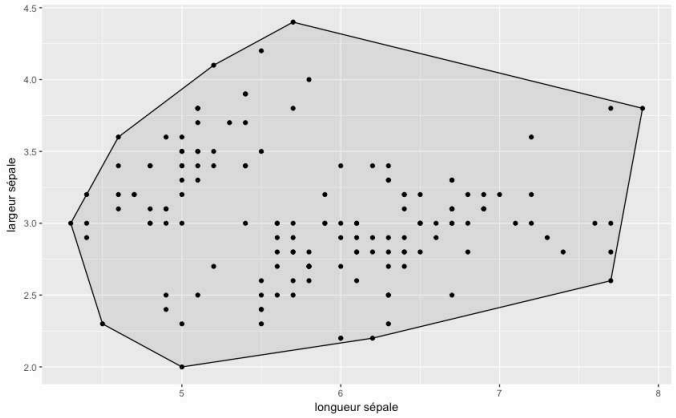



FIGURE 6.1 – Enveloppe convexe d'un ensemble de points dans un espace à 2 dimensions pour le jeu de données Iris Plants

Formellement par définition, si nous considérons $E$ un espace vectoriel, et $A$ une partie de $E$, l'enveloppe convexe de $A$ est l'intersection des parties convexes contenant $A$. C'est elle-même un convexe, et c'est le plus petit convexe contenant $A$. Le calcul de l'enveloppe convexe d'un ensemble de points est un problème classique en géométrie algorithmique. Il existe plusieurs algorithmes pour résoudre ce problème, la marche de Jarvis [1973], l'algorithme de Chan [1996] ou le parcours de Graham [1972]. Ces algorithmes fonctionnent dans des espaces à 2 dimensions et peuvent aussi fonctionner pour certains dans des espaces à 3 dimensions.

Comme pour l'enveloppe convexe, nous cherchons à retrouver les articles qui constituent l'enveloppe du profil d'un utilisateur (c'est-à-dire les articles qu'il a consommés). Le profil d'un utilisateur dans notre cas est composé de vecteurs à $n$ dimensions avec $n > 2$. Nous posons aussi la condition que le profil d'un utilisateur est composé de plus de 5 vecteurs correspondant aux vecteurs de représentation des articles. Nous ne pouvons pas appliquer les algorithmes de Jarvis, de Chan et de Graham dans notre cas. Notre objectif est de trouver une liste de $k$ vecteurs provenant du profil de l'utilisateur et pouvant représenter la diversité de ce profil. Pour trouver ces éléments, nous avons proposé l'algorithme 4. Nous avons initialisé la liste avec les éléments les plus différents du profil de l'utilisateur. Nous ajoutons à la liste dans un processus itératif un autre article *fp* (l'article le plus éloigné), qui est le plus différent de tous les articles déjà sélectionnés, c'est-à-dire la distance minimale entre cet article *fp* et tous les articles déjà sélectionnés est plus grande que celle des articles qui n'ont pas encore été sélectionnés.

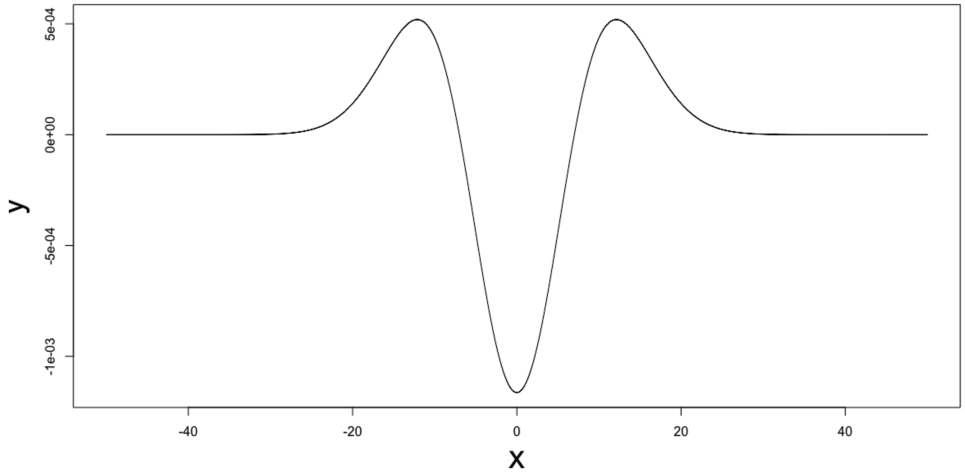
### 6.2.3 La fonction du Chapeau mexicain

Une fois que la diversité du profil de l'utilisateur a été identifiée et que nous avons produit une liste d'articles sources, il est nécessaire de faire couler ces sources et proposer des chemins

**Algorithme 4 :** Recherche des articles divers dans le profil d'un utilisateur

**Data** : $Dist$ : Matrice de distance ($n \times n$), $k$ : nombre d'articles différents souhaité, $P_\alpha$ profil de l'utilisateur $\alpha$
**Result** : $Gr_\alpha$ : La liste diversifiée d'articles sources
**begin**
  $Gr_\alpha = \{\}$
  **if** $k = 1$ **then**
    Sélectionner aléatoirement $i'$ tel que $i' \in P_\alpha$
    $Gr_\alpha = Gr_\alpha \cup \{i'\}$
  **else**
    $i_a, i_b = \max(Dist)$
    $Gr_\alpha = \{i_a, i_b\}$
    **while** $|Gr_\alpha| < k$ **do**
      $dfp = 0$ /* distance du point le plus éloigné */
      **for** $i_a \in P_\alpha - Gr_\alpha$ **do**
        $min_a = +\infty$
        **for** $i_b \in Gr_\alpha$ **do**
          **if** $Dist[i_a, i_b] < min_a$ **then**
            $min_a = Dist[i_a, i_b]$
        **if** $min_a > dfp$ **then**
          $dfp = min_a$
          $fp = i_a$
      $Gr_\alpha = Gr_\alpha \cup \{fp\}$

FIGURE 6.2 – Ondelette du Chapeau mexicain.




de découvertes. Les articles sources $Gr_u$ sont utilisés pour définir et produire une liste de recommandations $S_u$. Chaque article de la liste est utilisé pour retrouver un autre article qui sera recommandé, et tous les éléments retrouvés seront agrégés pour constituer la liste $S_u$. Pour retrouver des articles correspondants, nous proposons d'utiliser la fonction « Chapeau mexicain » qui possède des propriétés intéressantes, que nous allons expliquer dans les parties suivantes.

Pour trouver une zone d'intérêt à partir d'une fonction de dissimilarité, nous proposons le modèle du « Chapeau mexicain » qui nous permet de délimiter les frontières des articles diversifiés proposés par une plateforme de *streaming* en ligne. Pour promouvoir la diversité sur les services culturels en ligne (par exemple, musique, livre, jeu vidéo, vidéo), nous faisons l'hypothèse qu'un service culturel donné permet une exploration intéressante de la diversité dans la mesure où le service culturel propose des articles suffisamment dissemblables d'un article de référence, c'est-à-dire dans le cas de la recommandation musicale, si la musique recommandée est suffisamment différente de la musique habituellement écoutée et appréciée par un auditeur donné. Par conséquent, nous devons trouver une fonction pour définir les critères d'établissement d'une zone d'intérêt avec les propriétés suivantes :

(i) la fonction doit renvoyer une valeur minimale (ou négative) lorsque la dissimilarité est nulle,

(ii) la fonction doit croitre lorsque la dissimilarité augmente jusqu'à une valeur maximale,

(iii) la fonction doit diminuer après cet optimum jusqu'à tendre vers zéro lorsque la dissimilarité tend vers l'infini,

(iv) la fonction doit dépendre de quelques paramètres, telle que la moyenne ou l'écart-type des valeurs de dissimilarité calculé sur l'ensemble des contenus disponible pour décrire les articles du service.

La fonction de Chapeau mexicain, telle que représentée sur la figure 6.2, possède ces quatre propriétés désirées. Le Chapeau mexicain est la dérivée seconde de la fonction de loi normale $\mathcal{N}(\mu, \sigma)$. La loi normale est une distribution de probabilité caractérisée par la moyenne de la distribution $\mu$ et l'écart-type $\sigma$. Connaissant la moyenne $\mu$ et l'écart-type $\sigma$, la densité de probabilité d'une distribution normale est donnée par la formule 6.9, et la dérivée seconde est donnée par la formule 6.10 avec $\mu = 0$.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \ \mathrm{e}^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{6.9}$$

$$f''(x) = \mathrm{e}^{-\frac{1}{2}\left(\frac{x}{\sigma}\right)^2} \ \left(\frac{-\sigma^2 + x^2}{\sigma^5\sqrt{2\pi}}\right) \tag{6.10}$$

Cette fonction (comme dans la figure 6.2) ressemble à un sombréro (un « Chapeau mexicain ») à l'envers et a été utilisée dans un large éventail d'applications, par exemple dans le traitement des données sismiques [Ricker, 1944], dans la vision sur ordinateur pour détecter les bords dans les images numériques [Marr et Hildreth, 1980], ou comme noyau d'activation dans les cartes autoorganisatrice de Kohonen (une classe de neurones artificiels utilisée pour étudier la répartition de données dans un espace à grandes dimensions) avec les valeurs positives au centre, et les valeurs négatives dans le voisinage [Kohonen, 2001].

La Fonction du Chapeau mexicain est symétrique par rapport à l'origine. Pour une distribution de valeurs de dissimilarité $X = x_1, \ldots, x_n$ (les valeurs de dissimilarité sont positives, et on peut s'assurer de la normalité de la distribution par un test comme celui de Shapiro-Wilk), la valeur de $f$" sera négative pour $x \simeq 0$ et minimale pour $x = 0$. La valeur de $f$" augmentera avec la valeur de $x$ jusqu'au moment où elle atteint un maximum, et diminuera pour tendre vers 0 quand la valeur de $x$ s'éloignera de l'origine. Pour connaitre le point où la fonction commence à décroitre, nous devons trouver l'extrémum. Ce point correspond à un zéro de la dérivée troisième de la loi normale. Ce maximum est donné par la droite $o = \sqrt{3} \times \sigma$ (en rouge sur la figure 6.3). L'intersection de cette ligne passant par le point $o = \sqrt{3} \times \sigma$ avec le côté positif du Chapeau mexicain est interprété dans notre modèle comme l'élément intermédiaire optimal. Cet article est considéré comme ayant la meilleure diversité possible : ni trop proche ni trop éloigné de l'objet comparé. Nous considérons cet extrémum comme le centre de la zone intermédiaire. Pour trouver les frontières de cette zone, nous prenons $o \pm \gamma$ avec les intervalles $\gamma = (\sqrt{3} - 1) \times \sigma$ (limite en vert sur la figure 6.3). Effectivement, $b_{\text{inf}} = (\sqrt{3} \times \sigma) - \gamma$ correspond au point où la fonction devient positive. Par conséquent, nous pouvons définir deux limites : la borne inférieure $b_{\text{inf}} = (\sqrt{3} \times \sigma) - \gamma$ et la borne supérieure $b_{\text{sup}} = (\sqrt{3} \times \sigma) + \gamma$.

Ce modèle se base sur des paramètres tels que l'écart-type et la moyenne d'une distribution de valeurs de dissimilarité par rapport à un élément donné. Une fois que nous avons les valeurs de dissimilarité, les autres paramètres sont faciles à obtenir. Dans un scénario de recommandation, cette méthode nous permet d'éliminer les objets proches et éloignés et nous permet de nous concentrer sur les objets intermédiaires aux caractéristiques légèrement différentes de celles des

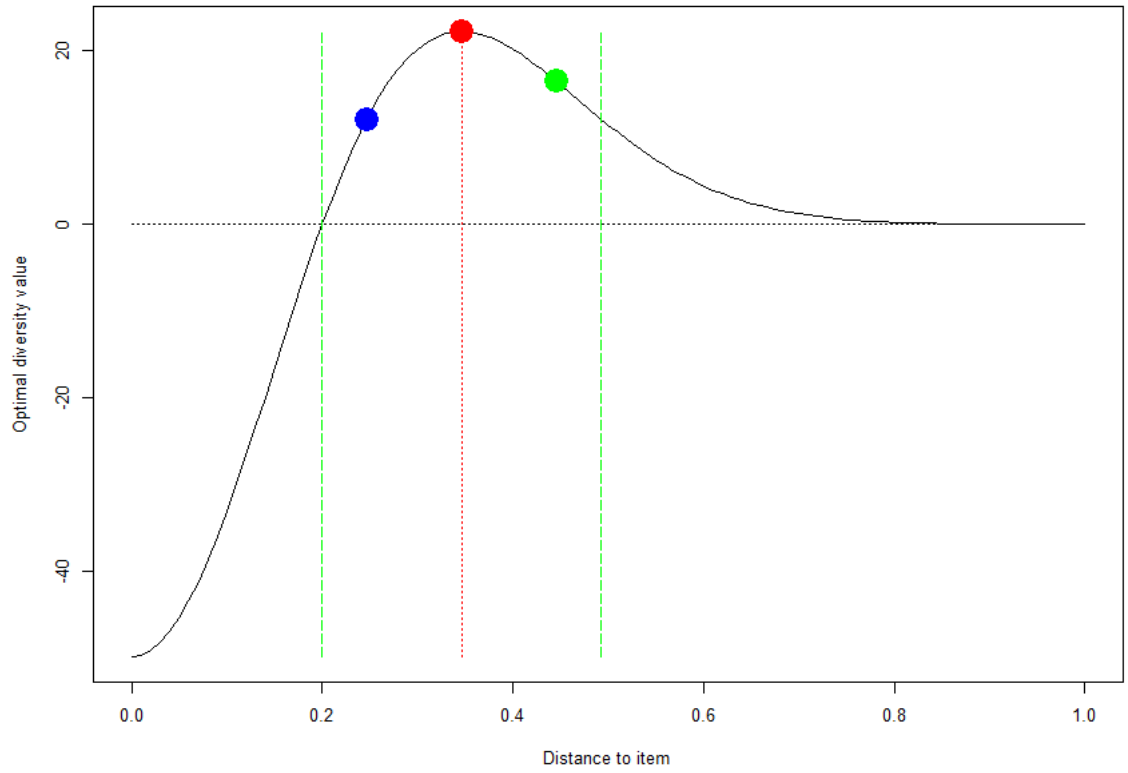



FIGURE 6.3 – Zone d'intérêt de diversité résultant de la Fonction du Chapeau mexicain : La distance à un élément de référence (source) est représentée sur l'axe X (axe des abscisses) et la diversité sur l'axe Y (axe des ordonnées). La zone de diversité optimale est située entre les deux droites vertes. La diversité optimale maximale est obtenue pour le point rouge.

objets que l'utilisateur consomme habituellement. Cette méthode est d'autant plus intéressante que plus nous avons de la diversité, plus la portée de la fonction est grande, plus la courbe est étendue.

Pour rechercher l'article optimal pour chaque article source, comme indiqué sur la Figure 6.3, l'algorithme 5 aura tendance à choisir pour les articles candidats ceux qui sont plus différents que l'article intermédiaire optimal (partie droite de l'intervalle) que ceux qui sont moins différents (partie gauche de l'intervalle). S'il n'y a pas d'élément à l'optimum, la recherche d'un objet candidat peut être effectuée sur la gauche de l'optimum (avec moins de différence) ou sur la partie droite de l'optimum (plus de différence). Cependant, pour la même distance, la fonction favorisera un élément situé sur le côté droit, c'est-à-dire être davantage éloigné de l'élément source (point vert sur la Figure 6.3), que sur le côté gauche, par exemple, le point bleu sur la Figure 6.3. Le point bleu a une valeur plus petite que le point vert même à distance comparable de l'optimum.

**Algorithme 5 :** L'algorithme de recommandation MHDM

**Data** : $P_\alpha$ : Ensemble des articles appréciés par un utilisateur $\alpha$, $Gr_\alpha$ : Ensemble des éléments sources du profil d'un utilisateur, $k$ : # nombre de recommandations, MHDM : dérivée seconde loi normale, sd : écart-type
**Result** : $S_\alpha$ : ensemble d'articles divers
**begin**
  $S_\alpha = \{\}$
  **for** $g$ *in* $Gr_\alpha$ **do**
    **if** $|S_\alpha| < k$ **then**
      $i^* = argmax_{i \notin P}\ MHDM(g, sd(I))$
      $S_\alpha = S_\alpha \cup \{i^*\}$
      $P_\alpha = P_\alpha \cup \{i^*\}$

## 6.3 Conclusion

Nous avons présenté ici deux familles d'algorithmes principalement basés sur le contenu qui ont pour objectif d'apporter de la diversité et de la nouveauté dans les listes de recommandation des utilisateurs. Ces algorithmes ont été développés pour répondre principalement au problème de la personnalisation accrue des systèmes de recommandation. Ces algorithmes ont un mécanisme semblable. Ils cherchent à modéliser les différents centres d'intérêt d'un utilisateur puis à partir de ces centres d'intérêt, ils créent des listes de recommandation en ajoutant une méthode de diversification de liste. Nous avons nommé respectivement ces deux familles d'algorithmes *Maximal Personalized Diversification Model* (MPDM) et *Mexican Hat Diversification Model* (MHDM). Le modèle MPDM utilise une méthode de clustering dans sa version simple et est associé à la méthode *Max-Sum Diversification* dans sa version gloutonne. Le modèle MHDM utilise une ondelette basée sur la loi normale pour modéliser une zone où l'utilisateur trouvera des éléments nouveaux. La diversité est apportée pour cet algorithme par la sélection d'éléments divers dans l'historique de l'utilisateur. Ces deux méthodes nécessitent des profils d'utilisateur composés de plusieurs articles. Mais ils peuvent aussi être utiles dans le cas d'un démarrage à fois surtout pour MHDM en ne prenant pas en compte la fonction de modélisation du profil de l'utilisateur. MPDM deviendra un algorithme classique de *Max Sum Diversification*, et MHDM n'utilisera que l'ondelette, c'est-à-dire la fonction « Chapeau mexicain », pour retrouver des produits nouveaux.

Dans le chapitre suivant, nous présenterons les expériences que nous avons menées avec ces algorithmes. Nous présenterons une évaluation que nous avons menée avec des sujets humains et aussi des évaluations hors ligne que nous avons menées sur des jeux de données ouverts. Nous comparerons nos résultats aux résultats obtenus par d'autres algorithmes de l'état de l'art, et nous ferons une analyse détaillée de ces résultats.

Chapitre 7

# Expérimentations

## 7.1 Évaluation d'un algorithme de calcul de distance entre des items musicaux

Dans cette section, nous allons présenter une expérimentation humaine que nous avons menée pour valider nos hypothèses sur le modèle dit du « Chapeau mexicain » (section 6.2.3) [Lhérisson *et al.*, 2017a]. Nous avons fait l'hypothèse de l'existence d'une zone intermédiaire qui se situe entre le trop similaire et le trop différent. Nous avons aussi fait l'hypothèse que dans cette zone, nous pouvons retrouver des produits qui sont différents et nouveaux pour un utilisateur par rapport à sa consommation passée. Avant la construction finale du système de recommandation nous avons pris le soin d'effectuer une « expérimentation humaine, » c'est-à-dire réaliser des études expérimentales avec des mesures formelles pour tester des théories sur la manière dont les utilisateurs vont interagir avec le système ou voir comment les utilisateurs vont percevoir certaines caractéristiques que nous voulons favoriser, comme dans notre cas les niveaux de différence. En introduisant le facteur humain au coeur de notre système, nous testons notre modèle basé sur le Chapeau mexicain, en calculant la dissimilarité entre des artistes et des morceaux de musique, et en vérifiant si le système est capable de proposer des morceaux divers par rapport à un morceau source. Ces morceaux divers proviennent de la zone intermédiaire, qui serait composée de morceaux et d'artistes qui, d'une part, ne sont ni trop semblables de ce que l'utilisateur connait déjà, et, d'autre part, qui ne sont ni trop éloignés sous peine de recommander des artistes complètement différents des intérêts musicaux de l'utilisateur. La difficulté consiste alors à définir l'écart convenable entre ces deux zones.

Nous allons présenter brièvement les mesures de similarité et de dissimilarité pour des items musicaux (section 7.1.1), puis présenter le modèle expérimental (section 7.1.2) et les résultats (section 7.1.3).

### 7.1.1 Similarité entre des morceaux de musique

Beaucoup d'études ont proposé plusieurs techniques pour calculer la similarité entre des articles musicaux, qu'il s'agisse de morceaux ou d'artistes [Gupta, 2014]. Pour mesurer cette similarité, plusieurs caractéristiques ont été utilisées. Certaines caractéristiques sont automatiquement extractibles en utilisant des algorithmes d'apprentissage automatique : caractéristiques

de bas-niveau (p. ex., La modulation par impulsion et codage), des caractéristiques de niveau intermédiaire (p. ex., spectrogramme) ou les caractéristiques de haut niveau (p. ex., les *tags*) [Brandenburg *et al.*, 2009]. Les caractéristiques de haut niveau permettent de faire le lien entre les concepts musicaux que l'utilisateur peut comprendre et ressentir et les concepts extraits de l'analyse du fichier audio [Schedl *et al.*, 2015]. Des méthodes ont été proposées pour taguer automatiquement les morceaux de musique, par exemple utiliser un réseau de neurones récurrents [Choi *et al.*, 2016].

Des annotations manuelles peuvent aussi être utilisées, comme des métadonnées éditoriales (c.-à-d., genre, sous-genre, label, année de diffusion, pays) pour alimenter un calcul de similarité. Des services en ligne comme *MusicBrainz*, *Discog* ou *Blitzr* peuvent être utilisés pour obtenir des données structurées créées par des experts [Magno et Sable, 2008]. Les annotations peuvent aussi venir de folksonomie comme *Last.fm* [Green *et al.*, 2009]. La similarité peut aussi être obtenue en utilisant les encyclopédies ouvertes comme *Wikipedia* ou les données liées ouvertes comme *DBpedia, BBC music* [Passant, 2010]. Quand les textes des chansons sont disponibles, ils peuvent aussi être utilisés comme caractéristiques pour la similarité [Lim *et al.*, 2013].

Pour résumer, la similarité entre des articles musicaux se base sur des informations textuelles reliées à la musique ou l'artiste, ou sur des caractéristiques acoustiques obtenues directement du fichier audio. Ces informations textuelles peuvent venir de sources externes ou peuvent être prédites à partir des caractéristiques acoustiques en utilisant des techniques d'apprentissage automatique.

Dans notre approche, nous nous sommes basés sur des informations qui nous viennent de services en ligne et d'*1D touch* la plateforme de streaming d'1D Lab. Notre objectif est d'utiliser dans nos calculs de similarité des concepts sémantiques compréhensibles par les utilisateurs. Nous avons utilisé l'API de *Blitzr* (une compagnie qui avait beaucoup de données éditorialisées sur la musique et qui a fermé en 2017 [1]) pour obtenir, quand ils étaient disponibles, des biographies des artistes ainsi que les genres musicaux des artistes. Nous avons aussi utilisé des *tags* prédits à partir du traitement du signal musical que nous avons récupéré de l'API de *Niland* (une compagnie qui avait un contrat avec 1D Lab et qui consistait à faire de l'analyse spectrale sur les musiques pour fournir des recommandations (compagnie qui a été rachetée par *Spotify* en 2017 [2]). Comme les genres musicaux jouent un rôle important dans notre mesure de similarité, nous avons mesuré la similarité entre ces genres en nous basant sur les liens qu'ils ont entre eux dans le graphe de *DBpedia* [Diefenbach *et al.*, 2016]. Notre mesure de dissimilarité est celle présentée dans le chapitre 5 à la section 5.3.

1. http://objectifaquitaine.latribune.fr/innovation/2017-10-02/blitzr-vie-et-mort-d-une-startup-751999.html
2. http://niland.io/

### 7.1.2 Protocole expérimental

Afin de vérifier que l'être humain peut percevoir différents niveaux de dissemblance, et aussi vérifier la pertinence de cette zone intermédiaire où la diversité optimale peut être trouvée (section 6.2.3), nous avons développé une application pour une expérimentation. Cette application fonctionne sur *Android* et *IOS*, et elle permet à l'utilisateur de suivre notre protocole.

Suivant les valeurs de distance que nous avons calculées, nous proposons aux utilisateurs, par rapport à un artiste musical de référence, des items musicaux à des distances variées de celui-ci : des items considérés comme « proches », d'une catégorie « intermédiaire » ou « différent ». L'expérimentation avec cette application a pour objectif d'évaluer la qualité de notre procédé. L'application est disponible sous le nom « AAPY » [3]

Sur cette application mobile, l'utilisateur écoute un item musical n°1, dit « item de référence », puis il écoute deux autres items musicaux n°2 et n°3 et il indique, à l'aide d'un curseur, la proximité qu'il ressent pour chacun des deux avec l'item n°1 de référence. Précisons qu'il ne s'agit pas pour l'utilisateur d'indiquer sa préférence entre des items musicaux mais bien de juger d'une proximité stylistique ressentie. L'exercice est répété avec le même item de référence et deux autres items musicaux n°4 et n°5. L'application permet de collecter la position du curseur pour chacune des évaluations ainsi que les titres musicaux qui ont été écoutés lors de l'exercice. Sachant que les items musicaux n°2, 3, 4 et 5 ont été piochés au hasard dans les classes d'items musicaux issus des catégories considérées par notre algorithme de calcul comme étant « proche », « intermédiaire » ou « différent » vis-à-vis de l'item de référence, l'exploitation des valeurs collectées nous permet de comparer ce procédé avec la perception de l'utilisateur. Les positions indiquées par les utilisateurs au moyen du curseur sont enregistrées et ces valeurs nous permettent de tester l'hypothèse que les items des classes « proche », « intermédiaire » et « différent » sont bien perçus comme tels par l'utilisateur. Nous démontrons aussi, grâce à notre procédé, que notre perception musicale basée sur les genres n'est pas que binaire et que nous sommes capables de sélectionner des items musicaux qui sont perçus au sens du genre musical comme n'étant ni vraiment proches ni vraiment distants d'un item musical donné.

L'architecture de cette application est la suivante. L'application est composée d'un serveur et d'une application cliente mobile. L'application cliente permet à l'utilisateur d'écouter des items musicaux et de situer les 2 items candidats vis-à-vis de l'item de référence à l'aide d'un curseur qu'il positionne entre « proche » et « différent » (cf. Figure 7.1). Le serveur comporte des parties algorithmiques et de stockage et d'échange de données (cf. Figure 7.2). Le serveur est relié au catalogue de musiques indépendantes d'*1D touch* [4] à partir duquel sont extraits les exercices en initiant l'item musical par *genre* ou *aléatoirement*. Les distances entre les genres [Diefenbach *et al.*, 2016], les artistes et les classes sont pré-calculées. Des API permettent au serveur de récupérer les résultats d'une recherche d'un utilisateur dans le catalogue mondial et des extraits

3. http://www.aapy.eu/
4. http://1dtouch.com/

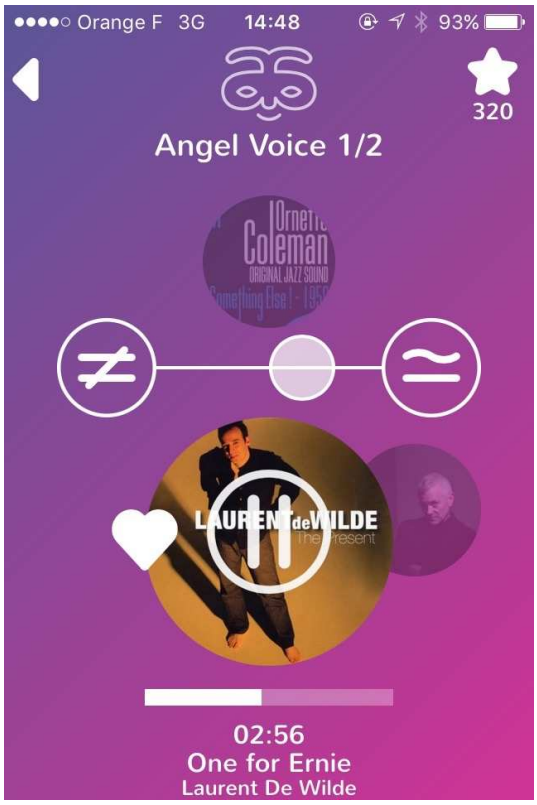



FIGURE 7.1 – Capture d'écran de l'application

sonores (*Niland*[5] et *Deezer*[6]). L'API *Blitzr*[7] permet de récupérer des informations descriptives sur l'artiste et l'album sélectionnés par l'utilisateur à partir de sa recherche. Ces informations sont transmises au module de calcul qui va les comparer à celles des artistes d'*1D touch* et déterminer, via les 2 bornes (section 6.2.3), les classes « proche », « intermédiaire » et « différent ».

### 7.1.3 Évaluation et présentation des résultats

Nous avons expérimenté notre application avec diverses personnes : les employés d'1D Lab (*7 personnes*), des spécialistes du secteur « musique » des bibliothèques (il s'agit ici des clients d'*1D Lab*), des étudiants, et d'autres personnes qui ont téléchargé l'application et qui ont fait les exercices de manière indépendante. Nous présentons ici les résultats obtenus par les étudiants, les employés d'1D Lab ainsi que les employés des bibliothèques. Il est facile de retrouver les catégories des différents utilisateurs ayant testé l'application expérimentale au moyen du type d'identifiant qui leur a été fourni. Nous avons collecté 322 jeux de comparaison à partir de 30 sujets.

Suivant les résultats obtenus par l'algorithme, pour un morceau de référence $r$, $i$ le morceau proche, $j$ le morceau intermédiaire, et $k$ le morceau différent, nous avons $d(r,i) < d(r,j) < d(r,k)$,

5. https://api.niland.io/doc/
6. https://developers.deezer.com/
7. https://blitzr.com/

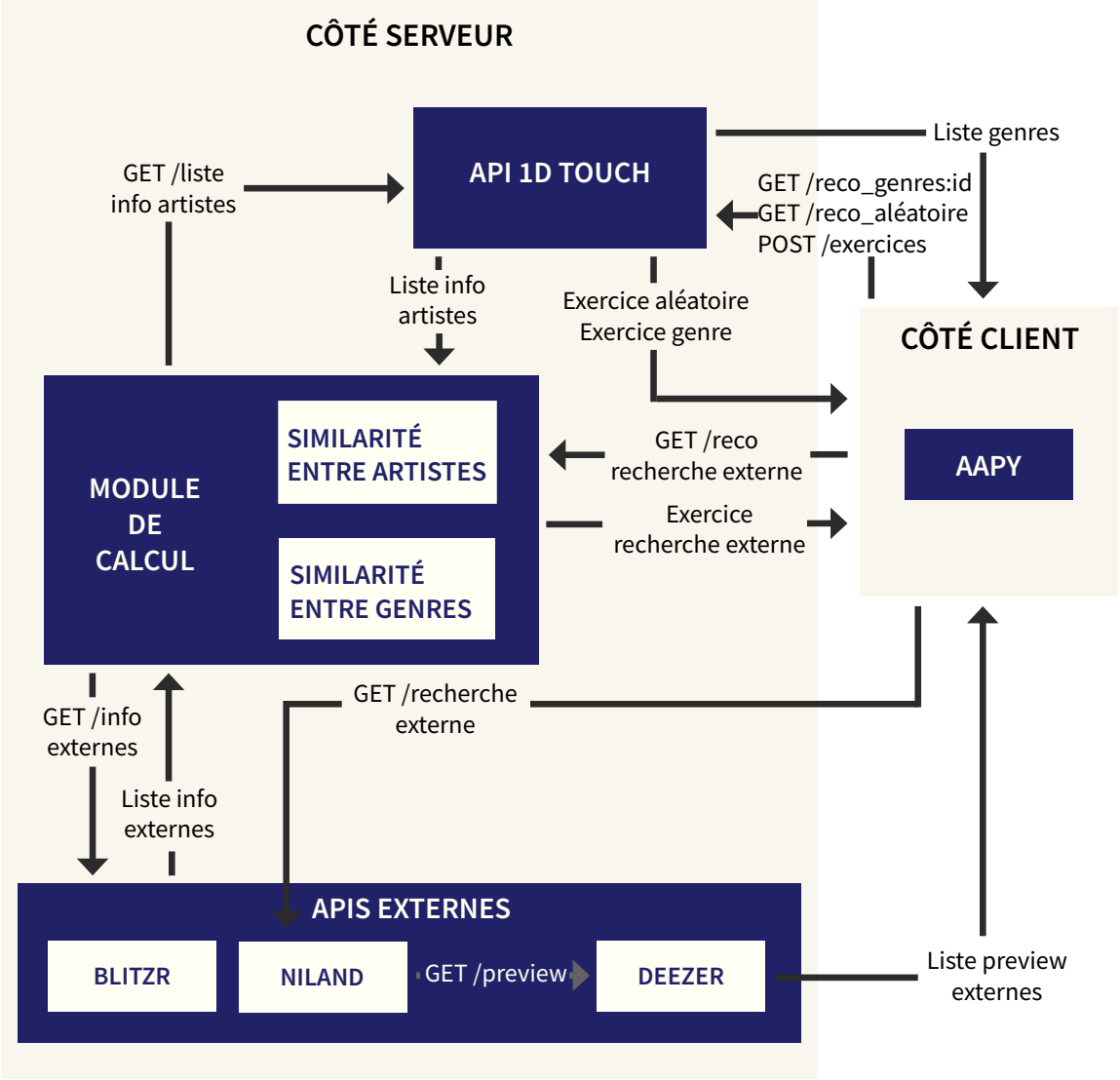



FIGURE 7.2 – Schéma de l'architecture de l'application

avec $d$ la fonction de dissimilarité (1 - similarité). Nous voulons vérifier si les utilisateurs perçoivent la dissimilarité $\delta$ entre les items dans le même sens que l'algorithme, $\delta(r,i) < \delta(r,j) < \delta(r,k)$. Nous calculons la moyenne des dissimilarités pour chaque classe.

Les résultats du tableau 7.1 montrent que cet ordre est respecté : $\delta(r,i) < \delta(r,j) < \delta(r,k)$, la moyenne de la distance perçue par les utilisateurs pour les items proches est plus petite que celle des intermédiaires qui est elle-même plus petite pour les items différents.

Pour voir si les utilisateurs perçoivent les éléments intermédiaires comme « pas trop » proches et « pas trop » différents par rapport à un item de référence, nous avons proposé deux types de comparaison. En fonction de l'item de référence, nous proposons un item de la classe « proche » contre un item de la classe « intermédiaire ». Dans un second test, nous proposons un autre couple d'items, un issu de la classe « différent » contre un de la classe « intermédiaire ».

Comme nous pouvons l'observer dans le tableau 7.2, nous observons que les utilisateurs

| | Moyenne | Écart-type |
|---|---|---|
| proche | 0.46 | 0.35 |
| intermédiaire | 0.60 | 0.35 |
| différent | 0.70 | 0.33 |

Tableau 7.1 – Dissimilarité (Moyenne) et écart-type des dissimilarités données par les utilisateurs pour les trois types de musique à comparer avec la musique de référence.

| | Moyenne | Écart-type |
|---|---|---|
| intermediaire vs. proche | 0.56 | 0.35 |
| intermediaire vs. différent | 0.65 | 0.35 |

Tableau 7.2 – Résultat des jeux de comparaison pour les deux types de test : item intermédiaire contre item proche, ou contre item de la zone différente.

ont tendance à donner à l'item intermédiaire une distance plus grande quand il est comparé à un item similaire que quand il est comparé à un item différent. Les utilisateurs accentuent les différences. Pour les musiques choisies par le modèle comme similaire à la musique de référence, la densité des réponses est importante pour les valeurs de dissimilarité minimales (proche de zéro) mais il y a aussi beaucoup de réponses pour lesquelles les valeurs de dissimilarité sont proches de 1. Pour les valeurs intermédiaires, la densité des résultats se trouve entre les résultats de densité similaires et les résultats de densité différents, les valeurs de dissimilarité données par les utilisateurs sont dispersées sur toute l'échelle, et moins concentrées sur les valeurs extrêmes (cf. Figure 7.3).

La différence perçue par les utilisateurs entre un item musical considéré similaire par le modèle et un item musical considéré comme différent par le modèle dans un *test de Student* bilatéral était statistiquement importante pour le risque de 5% (p-value 1.1e-04), aussi bien que la différence perçue par les utilisateurs entre un item considéré similaire par le modèle et un item musical appartenant à la zone intermédiaire (p-value 5.7e-03). Nous ne pouvons pas conclure qu'il existe une différence entre les évaluations des intermédiaires et des différents (p-value 7.9e-02 > 5e-02). En revanche, si nous considérons seulement les résultats des items intermédiaires qui ont été comparés aux items différents (nous prenons les tests intermédiaire vs. différent), la différence est importante statistiquement (p-value 2.4e-02 < 5e-02).

Par conséquent les résultats de l'expérimentation basée sur les utilisateurs corrobore notre hypothèse que l'être humain a les capacités d'identifier une zone intermédiaire qui est entre le trop similaire et le trop différent pour les items musicaux. Cela renforce l'idée qu'un modèle basé sur la dissimilarité peut être pertinent pour promouvoir la diversité et l'exploration dans les systèmes de recommandation. Les humains sont capables de percevoir les nuances de diversité. La présence de cette zone intermédiaire où la dissimilarité optimale peut être trouvée est présente

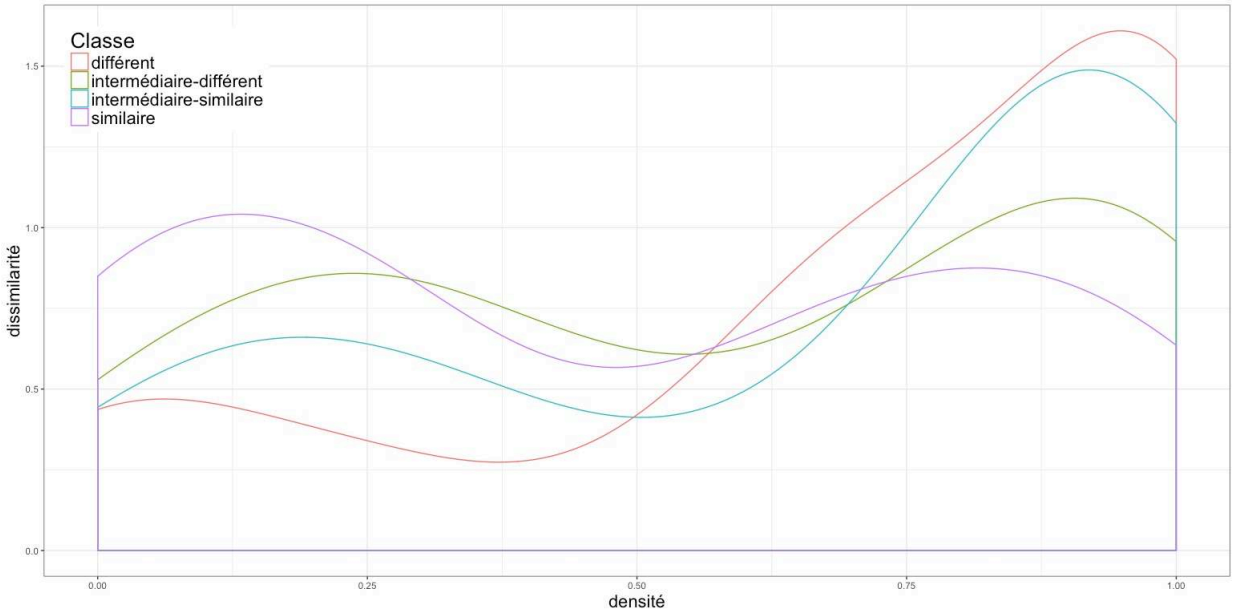



FIGURE 7.3 – Nombre de dissimilarité donné par les utilisateurs selon les classes calculés par le modèle.

pour la musique. Nous faisons l'hypothèse que cette perception peut être généralisée pour tous les biens culturels tels que le cinéma, la vidéo, etc.

## 7.2 Expérimentations hors ligne

Nous introduisons ici le cadre expérimental que nous avons mis en place pour valider les algorithmes de recommandation dont l'objectif est de promouvoir la diversité et la nouveauté sans importuner l'utilisateur. Nous suivons un protocole expérimental qui est commun aux différents travaux de recherche sur les systèmes de recommandation. Dans la suite de cette section, nous allons présenter dans la section 7.2.1 les jeux de données que nous avons utilisés pour valider nos approches. Nous allons présenter les algorithmes de l'état de l'art que nous avons utilisés comme référentiel auquel nous comparer dans la section 7.2.2. Nous finirons par présenter les méthodologies d'évaluation hors-ligne que nous suivrons dans toutes nos expérimentations dans la section 7.2.3.

### 7.2.1 Jeux de données

Nous avons sélectionné deux jeux de données utilisées dans l'état de l'art : *MovieLens100k*, *LastFM* et le jeu de données d'*1D touch*. Ces jeux de données nous permettent de tester nos algorithmes sur deux champs de recommandation : le cinéma et la musique. Deux de ces jeux de données sont disponibles en ligne, et le troisième est un jeu de données privé qui appartient à *1D Lab*. Les jeux de données *MovieLens100k* et *LastFM* comportent peu de données et ils

permettront à d'autres de reproduire nos expérimentations. Le jeu de données d'*1D touch* est de taille moyenne.

Nous présentons en détail ces jeux de données dans cette section et présentons une synthèse au tableau 7.3. Nous présentons leurs caractéristiques, le nombre d'utilisateurs ($|U|$), le nombre d'articles ($|I|$) ainsi que le type de l'article, le biais de popularité, la distribution des écoutes pour les données sur la musique ($|R|$) et des visionnages pour les données sur le cinéma ($|R|$). Comme nos algorithmes se basent sur le contenu, nous présentons aussi dans cette section les genres ($|G|$), et les *tags* ($|T|$) utilisés pour décrire les articles ainsi que le nombre d'items tagués ($|I_t|$) et le nombre d'items caractérisés par un genre ($|I_g|$). Pour les jeux de données *MovieLens100k* et *Last.fm* que nous avons utilisés, les genres et les *tags* sont livrés avec les jeux de données. Pour le jeu de données d'*1D touch*, nous avons utilisé des *tags* fournis par le partenaire d'*1D Lab*, *Niland*.

#### 7.2.1.1 *MovieLens100k*

Le jeu de données *MovieLens100k* [Harper et Konstan, 2016] est le plus petit des jeux de données *MovieLens* proposées par *GroupLens*. Nous avons récupéré la dernière version disponible de ce jeu de données (généré le 17 octobre 2016). Il contient 100 004 évaluations, 9 125 films ($|I|$), 671 utilisateurs ($|U|$), 18 genres différents, et 1 296 annotations faites par les utilisateurs qui donnent une liste de 592 *tags* uniques. Les évaluations varient de 1 à 5 et ont été produites par des utilisateurs, choisis aléatoirement, du site *movielens.org* [8] entre le 9 janvier 2009 et le 16 octobre 2016. Chaque utilisateur a évalué 20 films dans ce jeu de données. Il n'y a pas d'information démographique au sujet des utilisateurs.

Ce jeu de données est avec le jeu de données *MovieLens1M* l'un des plus utilisés dans la littérature pour tester les systèmes de recommandation. Il a l'avantage d'être petit, permettant des calculs rapides peu gourmands en ressources. Les résultats restent pertinents grâce à la densité et à la qualité des données. Il nous permet de prototyper rapidement nos systèmes et de nous comparer aux autres méthodes de l'état de l'art.

#### 7.2.1.2 *Last.fm*

Nous avons aussi testé nos algorithmes sur un jeu de données issu de *Last.fm*. Il avait été publié à l'occasion d'un atelier à la conférence *RecSys* 2011 [Cantador *et al.*, 2011] sous le nom *hetrec2011-lastfm-2k*. Ce jeu de données contient des informations sociales, des tags, ainsi que les écoutes des utilisateurs. Ce jeu de données est plus conséquent que le jeu de données de *MovieLens100k* en ce qui concerne le nombre d'utilisateurs et le nombre d'articles. Il contient 1 892 utilisateurs, 17 632 artistes musicaux. Il y a 92 834 relations utilisateur - écoute - artiste, 186 479 annotations produisant une liste finale de 11 946 *tags* unique. Dans ce jeu de données,

8. `http://movielens.org`

nous n'avons pas d'évaluations à notre disposition. Nous avons le nombre de fois qu'un utilisateur a écouté un artiste. Les données d'écoute sont moins denses que les données de *MovieLens*.

Ce jeu de données est très peu utilisé dans la littérature. Il a été utilisé pour faire des recommandations hybrides qui mélangent les données de *tags* et les écoutes. Il contient beaucoup de *tags*, ce qui nous permettra d'utiliser des algorithmes basés sur le contenu.

#### 7.2.1.3 *1D touch*

Le jeu de données *1D touch* est le jeu de données le plus récent que nous utilisons. Il contient les écoutes des utilisateurs effectuées entre le 28 février 2015 et le 1er octobre 2017.

Sur les 1 500 000 musiques disponibles sur *1D touch*, nous avons récupéré 1 304 896 morceaux tagués. Nous avons 53 genres pour qualifier nos morceaux. Pour les écoutes, nous avons constaté 20 193 utilisateurs uniques, 287 567 artistes écoutés, qui donnent 1 028 101 relations utilisateur - écoute - artiste. Contrairement aux autres jeux de données, le nombre de *tags* disponible sur *1D Lab* est beaucoup plus petit.

Ce jeu de données est privé. Il appartient à *1D Lab*, et n'est pas disponible en ligne. Ce jeu de données contient relativement peu d'écoutes comparativement au nombre d'écoutes générées sur *Spotify*, *Deezer*, ou *Pandora*. Le nombre de *tags* n'est pas élevé, mais nous permet quand même de produire des recommandations sur le contenu.

| | $\|R\|$ | $\|U\|$ | $\|I\|$ | densité | $\|G\|$ | $\|T\|$ | $\|I_g\|$ | $\|I_t\|$ |
|---|---|---|---|---|---|---|---|---|
| *MovieLens* | 100 004 | 671 | 9 066 | 1.64% | 18 | 592 | 9 125 | 689 |
| *Last.fm* | 92 834 | 1 892 | 17 632 | 0.27% | 880 | 11 946 | 6 772 | 12 523 |
| *1D touch* | 1 028 101 | 20 193 | 287 567 | 0.01% | 22 | 53 | 1 304 896 | 1 304 896 |

Tableau 7.3 – Caratéristiques des jeux de données utilisés dans nos expérimentations.

### 7.2.2 Méthodes de référence pour le reclassement

Dans les chapitres précédents (chapitres 2 et 3), nous avons abordé différentes techniques qui permettent d'apporter des éléments nouveaux, diversifiés dans les listes de recommandations. Ces ensembles de techniques nommées *beyond accuracy* en anglais, sont différentes approches qui ont été proposées pour générer des listes de recommandations pertinentes, mais aussi surprenantes et nouvelles pour les utilisateurs à qui elles sont adressées. Un ensemble de ces approches repose sur un reclassement des listes de résultats des algorithmes de l'état de l'art [Smyth et McClave, 2001; Ziegler *et al.*, 2005; Adomavicius et Kwon, 2014]. Un autre ensemble de ces approches reposent sur de nouveaux modèles avec des fonctions qui maximisent simultanément la pertinence et d'autres objectifs [Vargas *et al.*, 2012; Su *et al.*, 2013].

Dans les expérimentations hors-lignes que nous avons menées, nous avons principalement adopté une stratégie de reclassement de liste de recommandation.

#### 7.2.2.1 Reclassement basé sur la diversité

Nous avons comparé nos algorithmes à deux méthodes de diversification implicites : la *Maximal Marginal Relevance* (MMR) [Smyth et McClave, 2001] et la *Max-Sum Diversification* (MSD) [Borodin *et al.*, 2012]. Ces deux méthodes utilisent une approche gloutonne pour créer la liste de recommandations en cherchant un compromis entre la pertinence d'un article et la quantité de diversité que cet article apporte par rapport aux articles déjà sélectionnés. Dans nos expérimentations, pour chaque produit, MMR fait la combinaison linéaire de la similarité entre le profil de l'utilisateur et le produit à insérer dans la liste et la similarité maximale entre le produit à insérer et tous les produits déjà sélectionnés (formule 7.1).

$$\text{MMR} = \max_{i \in I \setminus P \cup S} \left( \lambda sim_1(u, i) - (1 - \lambda) \max_{j \in S} sim_2(i, j) \right) \tag{7.1}$$

La méthode MSD est aussi une combinaison linéaire d'une fonction qui mesure la pertinence d'un article pour un utilisateur et une fonction qui mesure la différence entre les articles sélectionnés (formule 7.2). Dans nos expérimentations, nous avons utilisé la similarité entre le profil d'un utilisateur et le produit à insérer dans la liste pour estimer la valeur de pertinence.

$$\text{MSD} = \underset{S \subseteq I \setminus I_u}{\text{argmax}} \left( \lambda sim(u, i) - (1 - \lambda) \sum_{i \in S} \sum_{j \in S - i} Dist(i, j) \right) \; tq \; |S| \leq k \tag{7.2}$$

Pour ces deux fonctions, nous utilisons pour la distance $dist$, la distance euclidienne comprise en [0, 1] avec : distance maximale = 1, distance minimale = 0. Nous utilisons pour les similarités, $sim_1$, $sim_2$ et $sim$, la similarité euclidienne : 1 - distance euclidienne. Cette valeur est aussi comprise entre [0, 1] avec : similarité maximale = 1, similarité minimale = 0. Ces deux fonctions objectives utilisent le paramètre $\lambda \in [0, 1]$ pour faire le compromis entre diversité et pertinence. Nous avons effectué des expérimentations pour $\lambda = 0.1$, $\lambda = 0.5$, $\lambda = 0.8$.

#### 7.2.2.2 Reclassement basé sur la nouveauté

Nous avons comparé nos méthodes aux méthodes qui apportent de la nouveauté dans les listes de recommandation. Pour la nouveauté, nous avons utilisé l'*Inverse User Frequency* (IUF) [Zhou *et al.*, 2010; Vargas et Castells, 2011]. L'IUF (formule 7.3) mesure le pourcentage d'utilisateur qui a interagi avec un article. Le logarithme permet de mettre l'accent sur les articles rares.

$$\text{IUF}(i) = -log_2(\frac{|\{u \in U, r_{ui} \neq \emptyset\}|}{|U|}) \tag{7.3}$$

Finalement, la fonction objectif est composée d'une fonction qui calcule la pertinence de l'article et une fonction qui calcule la nouveauté de cet article (équation 7.4) [Kaminskas et Bridge, 2017]. Nous reprenons pour notre expérimentation $sim_1$ (1 - distance euclidienne) et le

paramètre $\lambda = 0.1$, $\lambda = 0.5$, $\lambda = 0.8$.

$$\text{MIUF} = \max_{i \in I \setminus P \cup S} \lambda sim_1(u, i) - (1 - \lambda) IUF(i) \tag{7.4}$$

#### 7.2.2.3 Reclassement basé sur la surprise (unexpectedness)

Nous avons comparé nos algorithmes à une méthode qui favorise la surprise dans les listes de recommandations [Kaminskas et Bridge, 2017]. Une recommandation est surprenante pour un utilisateur si elle est différente des articles vus précédemment par l'utilisateur. Ces méthodes sont proches des méthodes qui font la promotion de la nouveauté.

La méthode que nous avons utilisée se base sur des probabilités de cooccurrence. Elle recommande à un utilisateur des articles qui ont peu de chances d'être vus en même temps en se basant sur la matrice d'utilité. Nous avons implémenté la méthode présentée par Kaminskas et Bridge [2017], qui utilise l'information mutuelle point par point normalisée (*point-wise mutual information*, PMI) de Bouma [2009] pour mesurer la probabilité d'observer deux variables indépendantes ensemble. Pour deux articles $i$, $j$ la valeur PMI est donnée par la formule 7.5.

$$\text{PMI}(i, j) = log_2 \frac{p(i, j)}{p(i)p(j)} / - log_2 p(i, j) \tag{7.5}$$

Les probabilités $p(i)$ et $p(j)$ sont les probabilités que les articles $i$ et $j$ soient vus par des utilisateurs. Elles sont données respectivement par les formules $\frac{|\{u \in U, r_{ui} \neq \emptyset\}|}{|U|}$ et $\frac{|\{u \in U, r_{uj} \neq \emptyset\}|}{|U|}$. La formule $p(i, j)$ est la probabilité qu'un utilisateur voie l'article $i$ et l'article $j$. Elle est donnée par la formule 7.6.

$$\text{p}(i, j) = \frac{|\{u \in U, r_{ui} \neq \emptyset \wedge r_{uj} \neq \emptyset\}|}{|U|} \tag{7.6}$$

La valeur de PMI est comprise entre $[-1, 1]$. Si PMI prend une valeur de -1, cela signifie que les deux articles, $i$ et $j$ n'ont jamais été vu ensemble. Si PMI prend une valeur de 0, cela signifie que les deux articles $i$ et $j$ sont indépendants, et une valeur égale à 1 signifie que les deux articles sont fortement corrélés.

Pour mesurer le niveau de surprise qu'un article apportera à un utilisateur, nous avons suivi la méthode employée par Kaminskas et Bridge [2017]. Nous calculons pour chaque article $i$ son PMI par rapport aux articles $j$ présents dans le profil d'un utilisateur (article que l'utilisateur a déjà vu). Nous prenons le complément des PMI normalisés entre $[0, 1]$ donnés par la formule 7.7. Nous prenons la valeur minimale donnée par ces PMI, et cette valeur nous renseigne sur la surprise minimale que l'utilisateur perçoit pour l'article $i$ quand ce dernier lui est recommandé [Kaminskas et Bridge, 2017].

$$\text{obj}_{\text{PMI}}(i) = \min_{j \in I \setminus P \cup S} \frac{1 - PMI(i, j)}{2} \tag{7.7}$$

#### 7.2.2.4 Algorithmes de l'état de l'art

Pour mesurer l'impact des méthodes de reclassement, nous avons implémenté dans un premier temps des algorithmes de l'état de l'art pour voir leur comportement en ce qui concerne la précision, la diversité, la nouveauté et la sérendipité. Nous avons utilisé dans un second temps ces algorithmes pour générer une liste de recommandations que nous avons ensuite reclassée en utilisant les méthodes de la section 7.2.2. Les algorithmes que nous avons implémentés sont les suivants : le *Bayesian Personalized Ranking* (BPR) [Rendle *et al.*, 2009], l'*Alternating Least Squares* (ALS) [Hu *et al.*, 2008], l'*Item-based collaborative filtering* (*item-based*) [Linden *et al.*, 2003], l'*User-based collaborative filtering* (*user-based*) [Herlocker *et al.*, 1999], le *content-based* [Pazzani et Billsus, 2007] et un algorithme basé sur la popularité le *popular-based* (*pop-based*). Ces algorithmes ont été présentés dans le chapitre de l'état de l'art (chapitre 1), et dans ce qui suit nous allons présenter les implémentations et les paramètres utilisés.

##### 7.2.2.4.1 Bayesian Personalized Ranking (BPR)

Nous avons utilisé la bibliothèque (*library*) *LigthFM* qui fournit des implémentations d'algorithmes de factorisation de matrice pour les systèmes de recommandation. Cette bibliothèque est proposée par Kula [2015]. Comme notre but n'était pas de maximiser la précision de nos algorithmes, nous avons utilisé les mêmes paramètres standards dans toutes nos expériences. Nous avons utilisé l'implémentation du *Bayesian Personalized Ranking* de cette bibliothèque qui maximise la différence de prédiction entre un exemple positif et un exemple négatif choisi au hasard. Cette méthode est utile lorsque seules les interactions positives sont présentes (c.-à-d. les articles que l'utilisateur a appréciés). Les paramètres que nous avons précisés sont la dimension des vecteurs latents (50), le nombre maximal d'exemples positif pour chaque itération (10) et l'exemple positif qui sera choisi parmi les exemples positifs à chaque itération (5). Nous avons aussi spécifié le nombre de pas (*epoch*) = 30.

**7.2.2.4.2 Alternating Least Squares** (ALS) Nous avons utilisé la bibliothèque python *implicit* qui offre une implémentation de l'algorithme *Alternating Least Squares* [Hu *et al.*, 2008]. Cette bibliothèque a l'avantage de faire les calculs de recommandation rapidement (60 000 fois plus vite que les autres bibliothèques qui implémentent le même algorithme selon son auteur). Les paramètres que nous avons définis sont la dimension des vecteurs latents (50), le paramètre de régularisation (0.01) et le nombre de pas (50).

**7.2.2.4.3 Item-based collaborative filtering - User-based collaborative filtering** Pour les deux algorithmes *item-based* et *user-based*, nous avons utilisé la bibliothèque *pyRecLab* de Sepulveda *et al.* [2017]. Le principal paramètre à définir pour ces deux algorithmes est le nombre de voisins. Nous avons fixé pour ces deux méthodes ce nombre à 100. L'autre paramètre à prendre en compte est la mesure de similarité. Nous avons opté pour la similarité de *Pearson*.

**7.2.2.4.4 Content-based** Nous avons implémenté cet algorithme en nous basant sur les méthodes définies dans l'état de l'art [Lops *et al.*, 2011; Baeza-Yates et Ribeiro-Neto, 1999]. Nous avons utilisé les *tags* et genres associés aux articles pour les décrire. Nous avons réduit les données en utilisant l'indexation sémantique latente (LSI). Pour chaque utilisateur dans la base de données, nous avons estimé la pertinence d'un article $i$ en mesurant la distance euclidienne de cet article $i$ par rapport à chaque article $j$ de son profil (articles déjà vus) et en sélectionnant l'article $j$ ayant la distance minimum par rapport à tous les articles du profil.

**7.2.2.4.5 Popular-based** (pop-based) Pour cet algorithme nous avons utilisé l'implémentation de la bibliothèque *pyRecLab* de Sepulveda *et al.* [2017]. Cet algorithme sélectionne les articles pertinents parmi les plus populaires pour un utilisateur.

Nous avons sélectionné ces algorithmes car ils couvrent l'ensemble des méthodes développées en système de recommandation : une méthode sur l'apprentissage de classement (*Bayesian Personalized Ranking*), une méthode issue des nombreuses méthodes développées lors du prix Netflix (*Alternating Least Squares*), deux méthodes historiques basées sur les $k$ plus proches voisins (*user-based collaborative filtering* et *item-based collaborative filtering method*) et une méthode basée sur le contenu. Les paramètres ont été choisis en fonction du temps de calcul, de l'espace qu'ils prenaient en mémoire, du résultat qu'ils donnaient et aussi de ce qui se faisait dans l'état de l'art.

À titre de comparaison, dans la littérature, le nombre de facteurs latents pour les algorithmes de factorisation de matrice est souvent choisi entre 25 et 100 [Jannach *et al.*, 2013; Bellogín *et al.*, 2013; Kaminskas et Bridge, 2017]. Le rythme d'apprentissage (*learning rate*) varie entre [0.01, 0.001, 0.0001] pour les algorithmes d'apprentissage de classement et de factorisation [Kaminskas et Bridge, 2017]. Le plus souvent, le nombre $k$ de voisins pour les algorithmes de filtrage collaboratif est compris entre 15 et 300 [Jannach *et al.*, 2013; Bellogín *et al.*, 2013; Kaminskas et Bridge, 2017]. Pampín *et al.* [2015] ont trouvé que la précision du *user-based collaborative filtering* s'améliorait jusqu'à $k = 90$ et restait constant jusqu'à $k = 200$. Pour la mesure de similarité utilisée, Gunawardana et Shani [2009] avaient montré que la similarité de Pearson ou du cosinus était les meilleures alternatives.

Pour tous les jeux de données, nous avons gardé les mêmes paramètres.

## 7.2.3 Évaluation des algorithmes de recommandation et de reclassement

### 7.2.3.1 Méthodologies d'évaluation

Nous avons mené deux ensembles d'expérimentations. Les premières expérimentations consistent à comparer les algorithmes de recommandation de l'état de l'art (section 7.2.2.4) sur le plan de la précision, du rappel, et de voir comment ces algorithmes se comportent naturellement en terme de diversité nouveauté, sérendipité, couverture. Les deuxièmes expérimentations consistent à appliquer les algorithmes de reclassement (sections 7.2.2.1, 7.2.2.2 et 7.2.2.3) sur

les résultats obtenus par les algorithmes de l'état de l'art. Pour le jeu de données d'1D touch nous nous sommes restreints volontairement à l'utilisation des résultats des algorithmes les plus performants lors de la deuxième phase.

Pour toutes les expérimentations, nous avons suivi le même protocole expérimental. Nous avons pour tous les jeux de données (*MovieLens, Last.fm, 1D touch*) utilisé une technique de validation croisée (la méthode *hold-out*). Nous avons pour chaque profil d'utilisateur pris 80% de ses interactions que nous avons placé dans l'ensemble d'entrainement (*train set*) et placé les 20% restant dans l'ensemble de test (*test set*). Tous les algorithmes ont appris sur le même ensemble d'entrainement et ont été testés sur le même ensemble *test*. Les ensembles d'entrainement nous ont permis de générer des recommandations pour les algorithmes de l'état de l'art présenté dans la section 7.2.2.4. Pour chacun de ces algorithmes, nous avons généré pour chaque utilisateur une liste de 100 recommandations que nous avons ensuite reclassées en utilisant les algorithmes de reclassement de l'état de l'art et les algorithmes que nous avons présentés dans le chapitre 6.

Nous avons ensuite utilisé des métriques issues de l'état de l'art (présentée à la section suivante) pour mesurer les performances des divers algorithmes utilisés et les comparer.

#### 7.2.3.2 Métriques d'évaluation

**Précision@k (precision)** La précision est la fraction d'articles pertinents retournés dans la liste d'articles de longueur $k$ recommandés avec $T_u$ les articles de l'ensemble test du $u^{ème}$ utilisateur.

$$\text{precision} = \frac{1}{|U|} \sum_{u \in U} \frac{1}{k} |R_u \cap T_u|$$

**Rappel@k (recall)** Il s'agit de la fraction d'articles pertinents qui sont dans la liste de recommandations des $k$ articles.

$$\text{recall} = \frac{1}{|U|} \sum_{u \in U} \frac{|R_u \cap T_u|}{T_u}$$

**Distributed Cumulative Gain@k (nDCG)** Normalized discounted cumulative gain@k est une mesure de pertinence graduelle qui prend en compte la pertinence des recommandations selon les articles se trouvant dans l'ensemble de test $T_u$ du $u^{ème}$ utilisateur, tout en donnant plus de poids aux articles se trouvant en tête de liste. La mesure $nDCG@k$ se calcule de la manière suivante, avec $r_i$ la valeur binaire de la pertinence de l'article $i$ dans la liste :

$$\text{nDCG} = \frac{1}{|U|} \cdot \frac{1}{iDCG@k} \sum_{u \in U} \sum_{i=1}^{k} \frac{2^{r_i} - 1}{log_2(1 + i)}$$

avec $iDCG@k$, le meilleur score obtenable par le $DCG@k$ (c.-à-d. dans le cas où tous les articles de la liste de recommandations sont dans l'ensemble test de l'utilisateur ). Les valeurs prises par $r_i$ sont 0 et 1, 0 si l'article n'est pas dans l'ensemble test et 1 s'il y est.

**Précision moyenne@k (aveP)** La précision moyenne est une mesure de précision graduelle, elle est donnée par la formule suivante :

$$\text{averageprecision} = \frac{\sum_{i=1}^{k} Precision \times r_i}{\sum r_i}$$

**Diversité de la liste@k (ild)** La diversité de liste (*intra-list diversity*) permet de mesurer la diversité d'une liste d'articles. Cette mesure a été proposée par Zhang et Hurley [2008] et permet de mesurer la distance moyenne entre toutes les paires d'articles dans la liste d'articles recommandée. Dans nos expérimentations, nous utilisons la distance euclidienne en nous basant sur le contenu des articles :

$$\text{ild} = \frac{1}{|U|} \sum_{u \in U} \frac{1}{k(k-1)} \sum_{(i,j) \in S_u} d(i,j)$$

**Sérendipité@k (unserendipity)** Nous nous sommes basés sur la formule de sérendipité présentée dans l'article de Zhang *et al.* [2012]. Elle permet de mesurer la surprise dans les recommandations en se basant sur la similarité entre le profil de l'utilisateur et la liste de recommandations. Nous nous sommes basés sur la métrique « Unserendipy », nous l'avons calculé en utilisant la distance euclidienne entre le profil de l'utilisateur $P_u$ et la liste de recommandations.

$$\text{unserendipity} = \frac{1}{|P_u|} \sum_{j \in P_u} \sum_{i \in S_u} \frac{d(i,j)}{k}$$

**Nouveauté@k (novelty)** La métrique de nouveauté que nous utilisons peut-être a été proposée dans l'article de Kaminskas et Bridge [2017]. Cette métrique de nouveauté se base sur l'entropie (*inverse user frequency*)

$$\text{novelty} = \frac{1}{nov_{max} \times |S| \sum_{i \in S} -log_2 \frac{|\{u \in U, r_{ui} \neq \emptyset\}|}{|U|}}$$

ou $U$ est l'ensemble des utilisateurs dans le jeu de données et $nov_{max} = -log_2 \frac{1}{|U|}$ est la valeur de nouveauté maximale, qui est utilisée pour normaliser le score de nouveauté de chaque utilisateur entre $[0, 1]$.

**Couverture du catalogue (coverage)** La couverture du catalogue nous permet d'évaluer le pourcentage d'articles dans le catalogue qui sont recommandés au moins une fois.

$$\text{coverage} = \frac{\left| \bigcup_{u \in U} S_u \right|}{|I|}$$

Nous avons utilisé des métriques qui couvrent toutes les dimensions que nous voulons tester dans nos algorithmes. La précision, le rappel et la précision moyenne (*averageprecision*) permettent d'avoir une idée sur la pertinence des recommandations. Ces mesures permettent de voir si les recommandations arrivent à atteindre les goûts des utilisateurs. La mesure de pertinence graduelle nous permet de voir comment les algorithmes rangent les listes de recommandations. Plus cette valeur est grande plus cela signifie que les articles pertinents sont présents en tête de liste. Les mesures de diversité (*ild*), nouveauté et de sérendipité permettent de voir comment les algorithmes arrivent à aller au-delà de la précision. Ces métriques nous servent aussi à mesurer l'impact du reclassement des listes sur ces dimensions. La couverture du catalogue nous permet de voir comment les algorithmes arrivent à diversifier les listes de recommandations.

## 7.3 Résultats

Nous présentons ici les résultats que nous avons obtenus à la suite des expérimentations hors ligne que nous avons effectuées. Dans la section 7.3.1, nous présentons les résultats des algorithmes de recommandation de l'état de l'art (section 7.2.2.4). Cette section comprend : la présentation des résultats des algorithmes sur les jeux de données *MovieLens* et *Last.fm* (section 7.3.1.1) et la présentation des résultats des algorithmes sur le jeu de données 1D touch (section 7.3.1.2).

Ensuite, dans la section suivante 7.3.2, nous présentons les résultats des algorithmes de reclassement. Nous étudions en section 7.3.2.1 la nouveauté et la diversité dans les profils des utilisateurs et nous analysons en parallèle l'impact du modèle MHDM (*Mexican Hat Diversification Model*) sur les profils des utilisateurs. En section 7.3.2.2 nous comparons les résultats obtenus par les algorithmes de reclassement aux résultats obtenus par nos algorithmes de reclassement, MHDM et MPD.

### 7.3.1 Étude des algorithmes de l'état de l'art

#### 7.3.1.1 Études des résultats sur *MovieLens* et *Last.fm*

Nous avons réalisé des expérimentations sur les jeux de données *MovieLens1k*, *Last.fm* en utilisant des algorithmes de recommandation tirés de l'état de l'art (section 7.2.2.4). Nous avons comparé les résultats de ces algorithmes selon les métriques présentées précédemment (section 7.2.3.2). Notre objectif est d'étudier le comportement de ces algorithmes en analysant leurs résultats pour ensuite choisir les algorithmes à implémenter pour le jeu de données 1D touch.

La figure 7.4 montre les résultats obtenus par chacun des algorithmes sur le jeu de données de *MovieLens1k*. Cette figure montre la moyenne des résultats des métriques pour des listes de longueur : $k$ = 5, 10, 20, 30, 40 et 50 items.

Sur le jeu de données *MovieLens1k*, BPR, ALS et *popular-based* ont des performances similaires. Ils obtiennent de meilleurs résultats sur la précision, le rappel et les métriques de

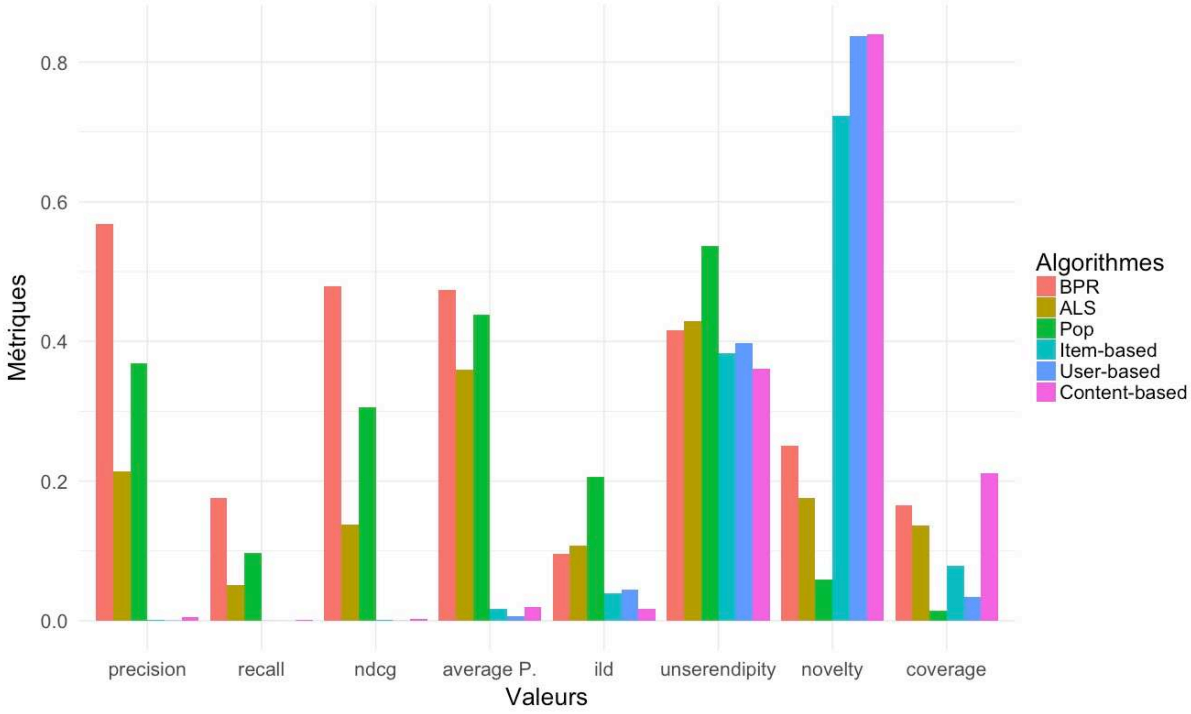



FIGURE 7.4 – Comparaison des résultats des algorithmes de l'état de l'art sur le jeu de données de *MovieLens100k*

classement (nDCG, Average Precision) que les algorithmes du filtrage collaboratif basés sur le voisinage (*user-based*, *item-based*). BPR qui apprend à classer les résultats réalise des meilleures performances sur la précision qu'ALS et *popular-based*. Les algorithmes BPR et ALS ont une meilleure couverture du catalogue que le *popular-based*, avec un léger avantage pour ALS. Mais BPR offre plus de nouveauté dans ses listes de recommandations qu'ALS et *popular-based* : ceci nous montre la propension qu'ALS a à se focaliser sur les produits les plus populaires. La forte valeur en précision de *popular-based* et sa faible valeur en couverture s'expliquent par les habitudes de visionnage de films des utilisateurs pour le jeu de données de *MovieLens1k*. Les utilisateurs ont regardé les mêmes films, et cet algorithme en ne recommandant que les produits populaires s'assure une forte précision, mais pénalise sa couverture de catalogue.

Les algorithmes basés sur le voisinage (*user-based*, *item-based*) et sur le contenu (*content-based*) ont de moins bons résultats de précision que le BPR, ALS et *popular-based*. Mais les algorithmes de filtrage collaboratif basé sur le voisinage et l'algorithme de filtrage basé sur le contenu apportent de la nouveauté aux utilisateurs. La couverture du catalogue réalisé par le filtrage collaboratif basé sur l'utilisateur est moindre que celle du filtrage collaboratif basé sur le produit. En revanche l'algorithme du filtrage basé sur le contenu couvre mieux le catalogue que les deux algorithmes basés sur le voisinage. Cet algorithme est le meilleur dans ce domaine dans nos expérimentations.

Tous les algorithmes sur ce jeu de données ont le même comportement pour la mesure de sérendipité (*unserendipity*). Pour des listes petites de recommandation (k=5, k=10), on

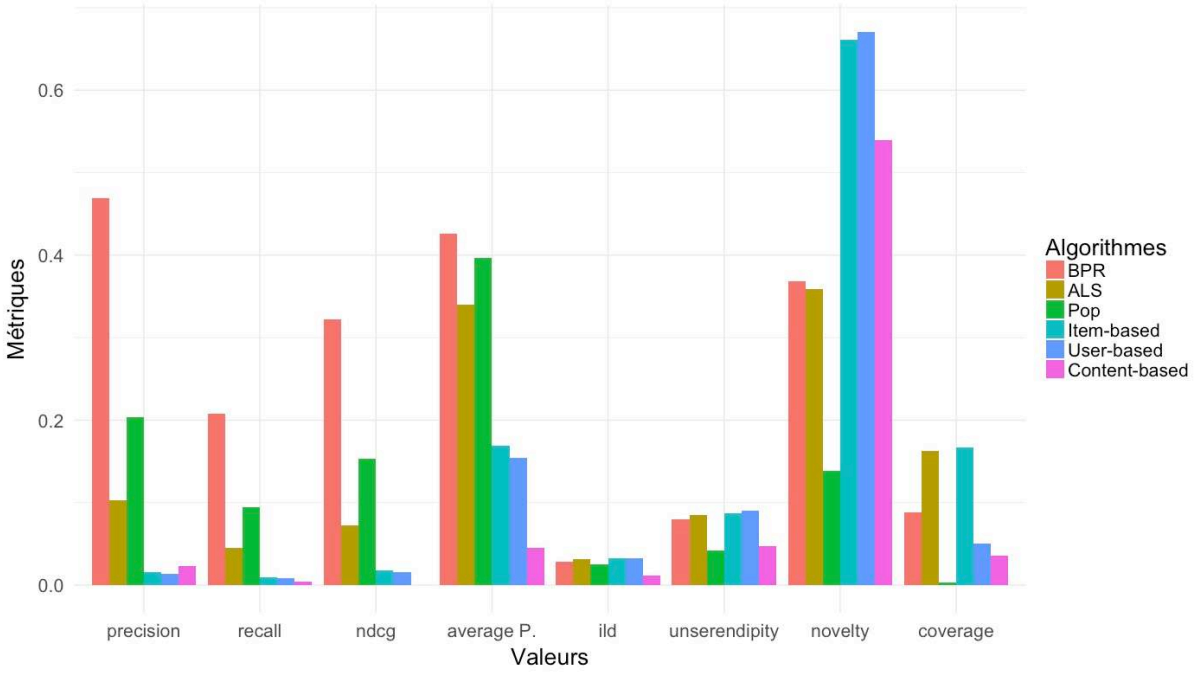



FIGURE 7.5 – Comparaison des résultats des algorithmes de l'état de l'art sur le jeu de données de Last.fm

remarque une très forte différence entre la nature des produits proposés et ceux déjà consommés par l'utilisateur (une grande valeur d'unserendipity), et cette valeur décroit quand k croit pour atteindre sa valeur minimale dans nos expérimentations à k=50 (tableau 7.4).

| | BPR | ALS | pop | item-based | user-based | content-based |
|---|---|---|---|---|---|---|
| Unserendipity@5 | 0.6727 | 0.6892 | 0.8112 | 0.7464 | 0.7736 | 0.7049 |
| Unserendipity@10 | 0.5838 | 0.6003 | 0.7445 | 0.6797 | 0.7069 | 0.6382 |
| Unserendipity@50 | 0.1868 | 0.1919 | 0.2458 | 0.1369 | 0.1408 | 0.1275 |

Tableau 7.4 – Résultats expérimentaux des algorithmes de l'état de l'art sur *MovieLens* avec des listes de recommandations de longueur $k$ = 5, 10, 50.

La figure 7.5 présente les résultats obtenus par les algorithmes de l'état de l'art sur le jeu de données *Last.fm*. Cette figure montre la moyenne des résultats des métriques pour des listes de longueurs $k$ = 5, 10, 20, 30, 40 et 50 items.

Sur le jeu de données *Lastfm*, on retrouve presque les mêmes résultats que sur celui de *MovieLens1k*. BPR, *popular-based* et ALS ont les meilleurs résultats de précision, de rappel et de classement. BPR apporte plus de nouveauté dans ses listes de recommandations qu'ALS et *popular-based*. Mais ALS est plus performant sur la couverture du catalogue que les deux autres algorithmes. Nous constatons que l'algorithme *popular-based* a le même comportement sur le jeu de données *Last.fm* que sur celui de *MovieLens*. Les utilisateurs n'ont écouté qu'une portion du catalogue, facilitant le travail de recommandation de *popular-based*. Ceci se traduit dans les chiffres obtenus par cet algorithme, une forte précision, mais une faible couverture.

Les algorithmes basés sur le filtrage collaboratif *item-based, user-based* et sur le contenu ont de piètres performances en matière de précision, rappel et classement sur le jeu de données de *Last.fm*. Mais en revanche, ces algorithmes produisent des listes avec un fort apport en nouveauté. Le filtrage basé sur le produit assure une meilleure couverture du catalogue que le filtrage basé sur l'utilisateur et le filtrage basé sur le contenu.

Pour la mesure de sérendipité, on retrouve dans l'ensemble le même comportement que sur le jeu de données de *MovieLens1k* : des valeurs qui décroissent quand $k$ croit. En revanche, les valeurs observées sur ce jeu de données sont plus petites que celles observées sur le jeu de données de *MovieLens1k*. On observe le problème de personnalisation accrue pour le filtrage basé sur le contenu, la valeur d'*unserendipity* obtenue par content-based est faible. L'algorithme ne propose pas aux utilisateurs des articles différents de ceux qu'il a déjà consommés.

Somme toute, sur ces deux jeux de données, nous pouvons distinguer deux groupes d'algorithmes. Le premier groupe, composé de BPR, ALS et *popular-based*, contient des algorithmes qui sont précis et arrivent à cibler les préférences des utilisateurs. En revanche, ces algorithmes n'ont pas les mêmes performances sur la diversité, la sérendipité, la nouveauté et la couverture du catalogue. Le second groupe, composé des algorithmes de filtrage collaboratif basé sur le voisinage (*user-based*, *item-based*) et sur le contenu, contient les algorithmes qui ne sont pas précis, mais offrent beaucoup de nouveauté dans leur liste de recommandations.

#### 7.3.1.2 Études des résultats sur 1D touch

Nous avons utilisé les algorithmes BPR et ALS sur le jeu de données 1D touch. Ce choix est motivé par la qualité des résultats (surtout pour les mesures de précision) observés sur ces algorithmes préalablement sur les jeux de données *MovieLens* et *Last.fm*. L'objectif est d'améliorer la nouveauté, la sérendipité, la diversité et la couverture dans les listes de recommandations produites par des algorithmes de l'état de l'art. Ceci va générer une perte en précision. Cela nous permettra d'observer les comportements de nos algorithmes. De plus, même si la précision ne fait pas tout dans un système de recommandation, cette métrique nous renseigne quand même sur la pertinence des listes de recommandations (l'algorithme arrive à cerner les préférences de l'utilisateur). Nous avons aussi fait le choix d'utiliser ALS et BPR pour leur vitesse de calcul. En effet, ces algorithmes, dans leurs versions implémentées dans les bibliothèques python *Implicit* et *LightFM*, construisent des modèles rapidement sur nos jeux de données et ne prennent pas beaucoup de temps pour produire les listes de recommandation (tableau 7.5). Pour l'algorithme *content-based*, nous n'avons pas utilisé de bibliothèque. Pour générer la liste de recommandations d'un utilisateur, nous avons calculé la distance entre son profil et chacun des articles qu'il n'a pas encore vus, puis nous avons classé par ordre croissant les résultats de distance obtenus.

| | ALS | BPR | item-Based | user-based | content-based |
|---|---|---|---|---|---|
| entrainement | 2.57 | 8.43 | 53.74 | 2.70 | 105.6* |
| recommandation | 1.08 | 1.43 | 18607.55 | 725.86 | |

Tableau 7.5 – Temps de calcul en secondes des algorithmes sur le jeu de données *MovieLens1k*

La figure 7.6 montre les performances des algorithmes BPR, ALS, *popular-based* sur le jeu de données 1D touch. Comme pour les résultats des algorithmes sur les jeux de données de *MovieLens* et *Last.fm*, nous avons fait ici aussi la moyenne des résultats des métriques pour des listes de longueur $k$ = 5, 10, 20, 30, 40 et 50 items.

On remarque que BPR obtient encore les meilleurs résultats en précision. Cette fois, l'écart est conséquent comparativement à ce qui a été observé sur MovieLens et Last.fm. Les valeurs de précision, de rappel et de classement sont plus petites que sur les autres jeux de données. L'algorithme ALS obtient de meilleurs résultats de précision que *popular-based* sur ce jeu de données, mais ne couvre pas mieux le catalogues que BPR. Les résultats en diversité sont très faibles surtout pour BPR et ALS, et les résultats en sérendipité suivent la même tendance que pour les autres jeux de données : on retrouve une grande valeur de sérendipité pour les petites listes de recommandations ($k$ = 5, 10) et cette valeur de sérendipité diminue considérablement quand $k$ augmente.

L'algorithme *popular-based* obtient par ailleurs les meilleures valeurs de sérendipité et de diversité. Le jeu de données d'1D Lab contient peu d'écoutes par rapport à sa taille, donc il y a peu de morceaux écoutés par beaucoup d'utilisateurs. Contrairement aux autres jeux de

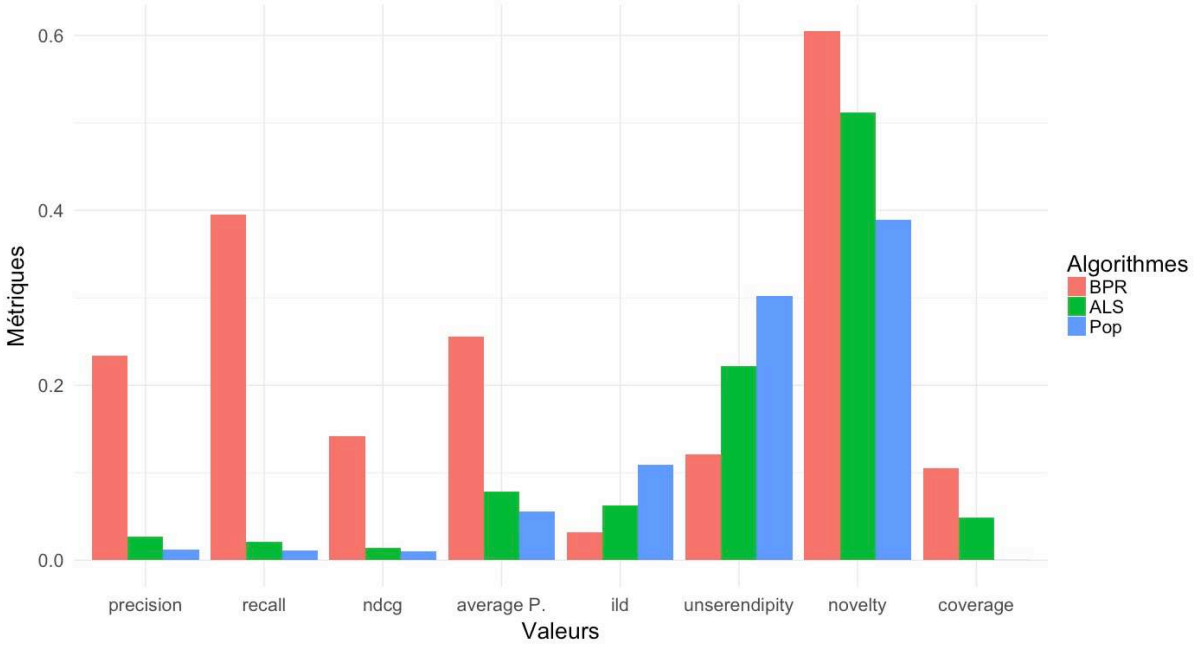



FIGURE 7.6 – Comparaison des résultats des algorithmes de l'état de l'art sur le jeu de données d'1D touch

données utilisés pour les expérimentations, dans le jeu de données d'1D touch, les articles les plus populaires sont écoutés par un petit nombre d'utilisateurs. Sur le jeu de données 1D touch, BPR est l'algorithme qui a les meilleures performances sur le plus de métriques. Dans la suite des travaux menés sur ce jeu de données, nous avons principalement utilisé BPR comme base pour les algorithmes de reclassement.

### 7.3.2 Au-delà de la précision

Dans cette section, nous allons présenter les résultats des algorithmes de reclassement sur une sélection des algorithmes de l'état de l'art (section 7.3.2.2). Parmi ces derniers, nous avons choisi l'algorithme BPR qui obtient les meilleurs résultats en précision, l'algorithme ALS qui obtient en règle générale les deuxièmes meilleurs résultats et finalement *item-based* car nous voulons aussi étudier le comportement des méthodes de reclassement sur un algorithme de recommandation qui obtient des résultats moins probants.

Mais nous allons d'abord étudier la diversité du profil de l'utilisateur (section 7.3.2.1) qui est pris en compte dans l'algorithme MHDM que nous avons développé (section 6.2).

#### 7.3.2.1 Diversité et nouveauté du profil de l'utilisateur

Ici nous allons présenter la diversité moyenne observée dans les profils des utilisateurs, et nous allons la comparer à la diversité moyenne des nouveaux profils des utilisateurs créés après avoir appliqué l'algorithme 4 du modèle MHDM (section 6.2) (figures 7.7, 7.9, 7.11). Nous allons parallèlement présenter la nouveauté observée dans les profils des utilisateurs et nous allons aussi la comparer à la nouvelle nouveauté des profils des utilisateurs après avoir utilisé l'algorithme 4 (figures 7.8, 7.10, 7.12).

L'algorithme 4 est la première étape du modèle MHDM (section 6.2). Il sélectionne les $k$ items les plus divers du profil de l'utilisateur. Nous appelons « profil initial de l'utilisateur » la liste des items avec lesquels l'utilisateur a interagi dans le passé. Pour mesurer la diversité entre les profils des utilisateurs, nous avons utilisé la métrique ild présentée dans la section 7.2.3.2, pour la nouveauté nous avons utilisé la métrique novelty présentée dans cette même section 7.2.3.2.

| | minimum | maximum | Q1 | mediane | Q3 | moyenne | écart-type |
|---|---|---|---|---|---|---|---|
| movie lens | 11 | 1894 | 29 | 57 | 127 | 119.22 | 184.4 |

Tableau 7.6 – Données statistiques des profils utilisateurs sur le jeux de données *MovieLens100k*

L'algorithme 4 du modèle MHDM arrive à extraire la diversité du profil de l'utilisateur sur le jeu de données *MovieLens100k*. La réduction du profil de l'utilisateur à un ensemble de $k$ = 5, 10 et 20 articles permet d'obtenir une diversité deux fois plus importante que celle du profil initial des utilisateurs. Le nouveau profil extrait pour les $k$>20 a aussi une diversité supérieure à celle du profil initial, puis cette diversité décroit pour ensuite être sensiblement égale à celle du profil initial à $k$ = 50. Les profils utilisateurs initiaux du jeu de données de *MovieLens* sont tous composés de plus de 10 films (tableau 7.6). En moyenne, les utilisateurs ont visionné (évalué) 119 articles (tableau 7.6). Dans ce jeu de données, le nombre de visionnages de films varie entre les utilisateurs. Statistiquement, la variable « nombre de films visionnés » est dispersée autour de la moyenne : l'*écart-type* est très élevé (tableau 7.6). Les utilisateurs ayant regardé plus de 50 films ont permis d'obtenir une diversité moyenne de profil d'utilisateur à $k$>30 supérieure à la diversité initiale (la valeur médiane du nombre de films regardés est = 57).

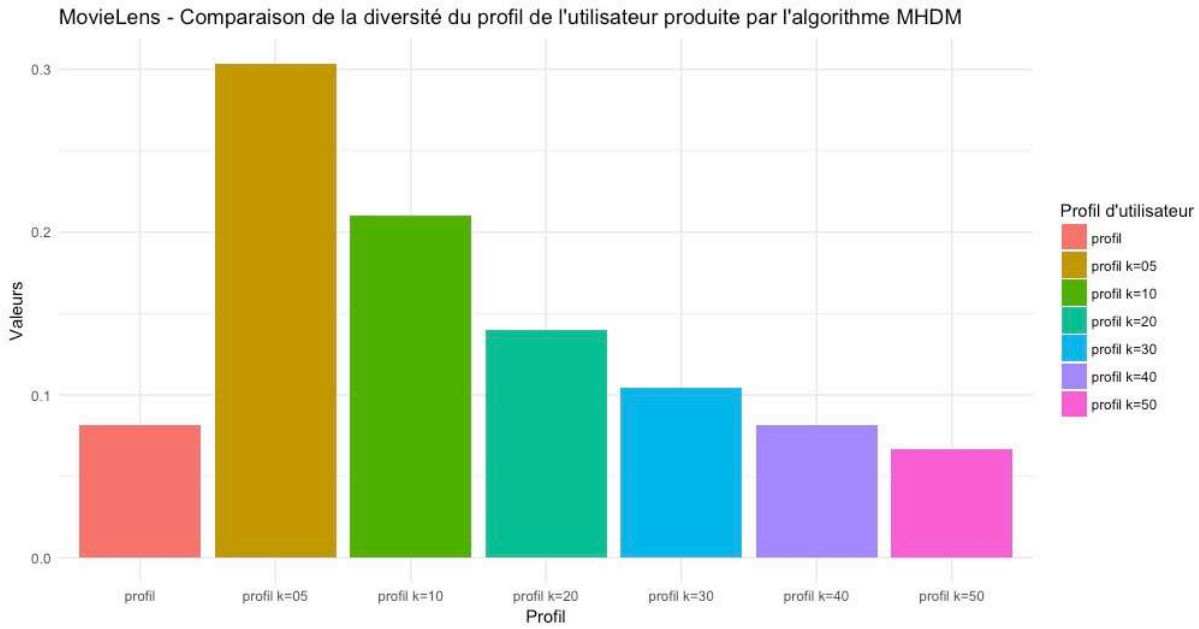



FIGURE 7.7 – Variation de l'*intra-list diversity* du profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *MovieLens*

La figure 7.8 nous montre la variation de la nouveauté dans les profils des utilisateurs suite à l'application de l'algorithme 4. La sélection de produits divers par l'algorithme 4 du modèle MHDM fait diminuer la nouveauté du profil de l'utilisateur. Cette nouveauté augmente avec k, et atteint un maximal à $k = 20$ puis diminue pour $k = 30, 40, 50$. Cette diminution de la nouveauté quand $k$ augmente nous montre que les utilisateurs ont visionné les mêmes films.

| | minimum | maximum | Q1 | mediane | Q3 | moyenne | écart-type |
|---|---|---|---|---|---|---|---|
| Last.fm | 1 | 48 | 38 | 40 | 42 | 39.27 | 5.37 |

Tableau 7.7 – Données statistiques des profils utilisateurs sur le jeu de données *Last.fm*

L'application de l'algorithme 4 sur le jeu de données *Last.fm* pour la sélection des produits les plus différents du profil d'un utilisateur est efficace pour $k$ allant de 5 à 30 (figure 7.9). Pour $k = 40$ la diversité est inférieure à celle des profils initiaux. Le nombre maximal d'articles qu'on retrouve dans le profil d'un utilisateur pour ce jeu de données est 48 avec une moyenne de 39 articles par profil (tableau 7.7). L'écart-type de la distribution du nombre d'articles dans les profils est faible (tableau 7.7). Les utilisateurs ont dans l'ensemble écouté entre 30 et 45 artistes. L'algorithme retrouve peu de profils avec plus de 40 articles (le troisième quartile est 42). À $k = 50$ l'algorithme retourne les profils initiaux, le nombre maximum d'artistes dans les profils est égal à 48.

La nouveauté présente dans les profils des utilisateurs de *Last.fm* est importante ($\simeq 0.5$)(Figure 7.10). Cette nouveauté diminue quand on sélectionne $k$ produits différents dans le profil de l'utilisateur. Pour $k = 5$ à $k = 30$ elle croit, et à $k = 40$ elle prend la plus petite valeur.

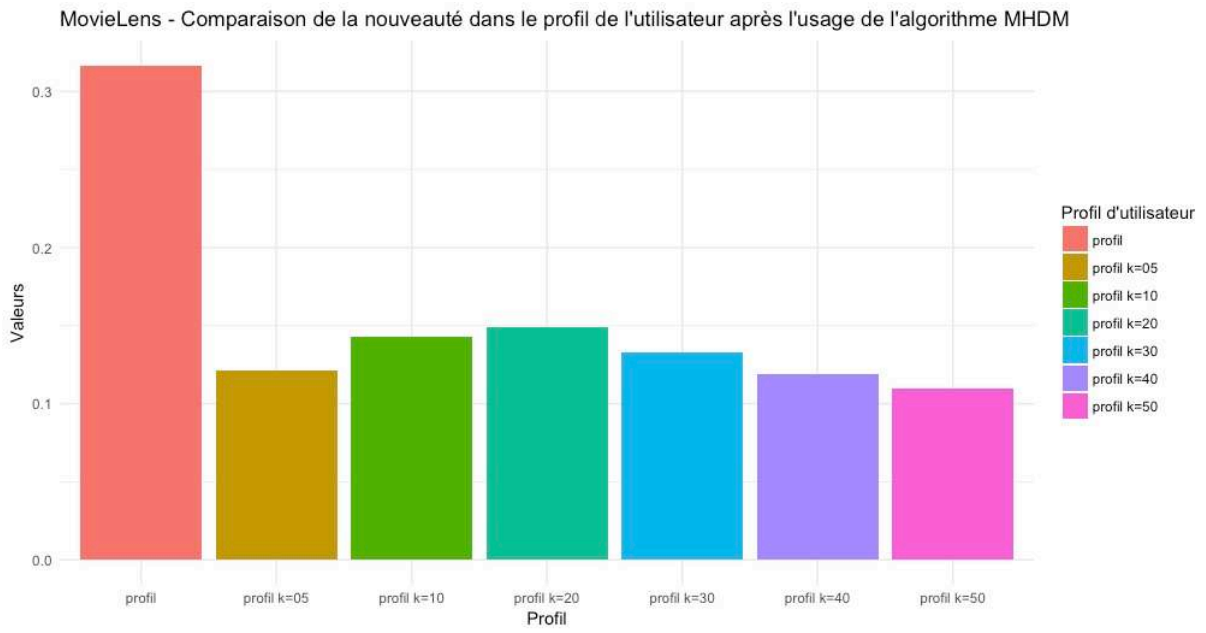



FIGURE 7.8 – Variation de la nouveauté *novelty* dans le profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *MovieLens*

| | minimum | maximum | Q1 | mediane | Q3 | moyenne | écart-type |
|---|---|---|---|---|---|---|---|
| 1D touch | 1 | 9891 | 2 | 6.0 | 20 | 43.42 | 216.99 |

Tableau 7.8 – Données statistiques des profils utilisateurs sur le jeu de données *1D touch*

Pour le jeu de données d'1D touch, l'algorithme 4 est efficace seulement pour $k = 5$. Pour cette valeur de $k$, on retrouve un nombre important de profils d'utilisateur de longueur supérieur à 5. Le jeu de données d'*1D touch* comprend un petit groupe d'utilisateurs ayant écouté plusieurs morceaux (les prescripteurs de contenu) et un groupe d'utilisateurs ayant écouté peu de morceaux (les utilisateurs normaux). Nous n'avons pas séparé ces deux groupes d'utilisateurs dans nos expérimentations. Cela donne une moyenne de 43 morceaux écoutés par utilisateurs, mais avec une variance et un écart-type importants. Les valeurs de nombre d'écoutes sont éparpillées autour de la moyenne. La valeur médiane du nombre d'artistes dans les profils utilisateur est égale à 6.

La variation de la nouveauté sur le jeu de données d'1D touch diffère de celle de *Last.fm*. Sur ce jeu de données, la nouveauté dans les listes d'écoute des utilisateurs est grande (supérieure à 0.6). Mais cette nouveauté diminue avec la taille des profils. Elle reste supérieure aux valeurs de nouveauté dans les profils des utilisateurs de *MovieLens100k* et est comparable à celles observées dans les profils des utilisateurs de *Last.fm*.

Ces données sur la nouveauté et la diversité des profils des utilisateurs seront mises en exergue dans la suite de l'analyse des résultats, notamment dans la section 7.3.2.2. Nous allons y présenter les résultats des algorithmes de reclassement de listes de recommandations obtenues préalablement par des algorithmes de l'état de l'art.

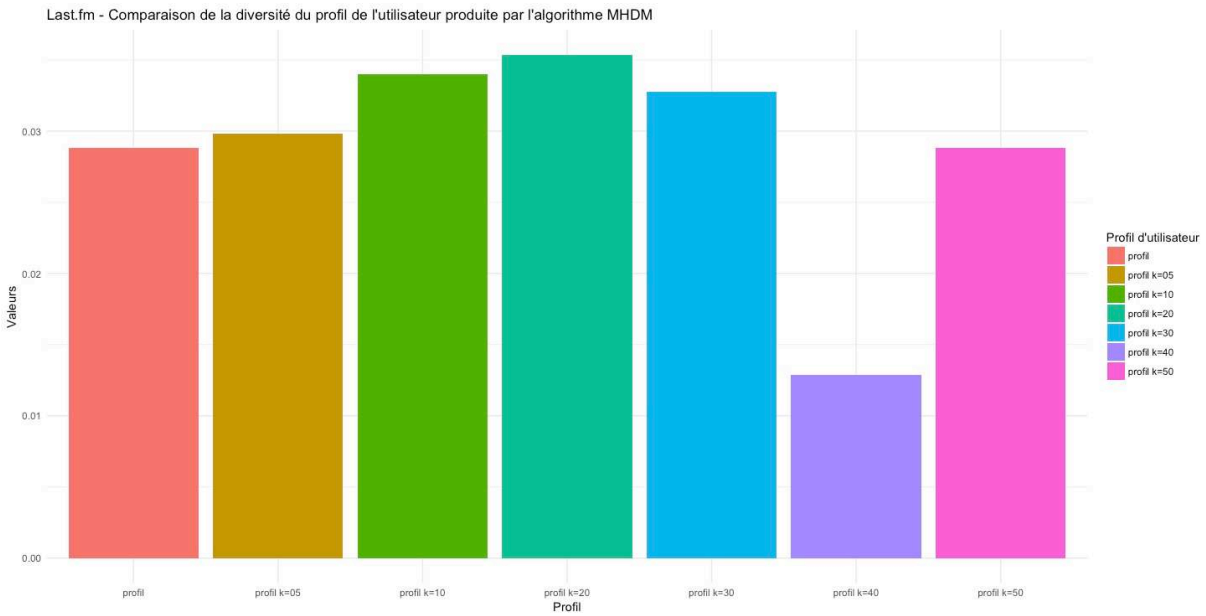



FIGURE 7.9 – Variation de l'*intra-list diversity* du profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *Last.fm*

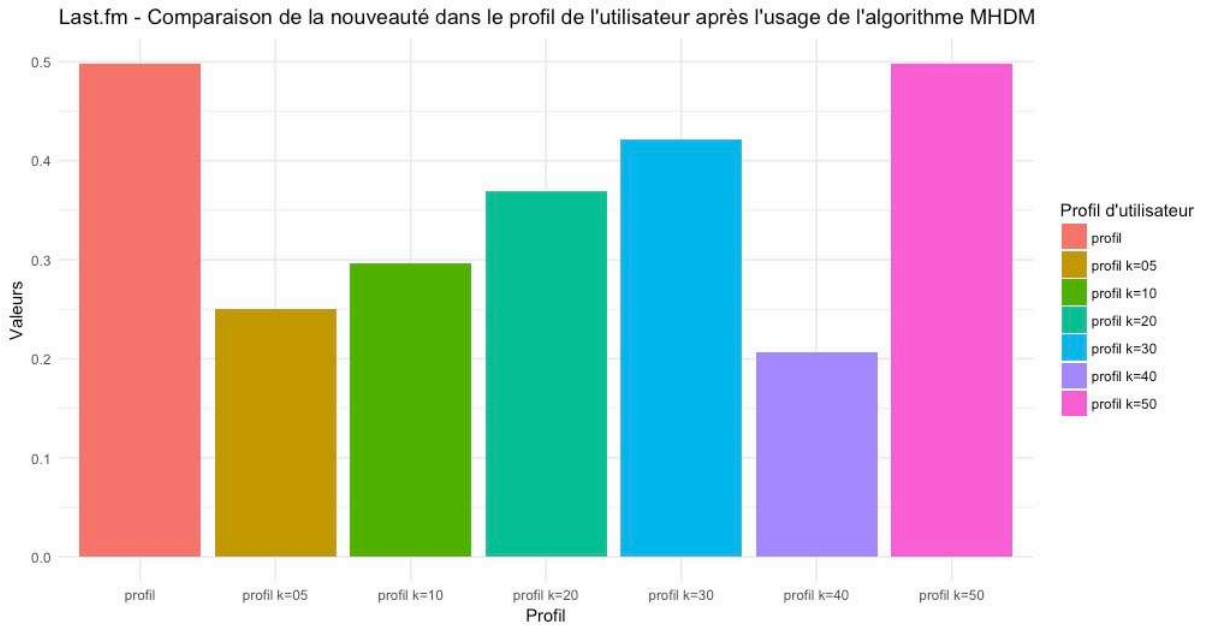



FIGURE 7.10 – Variation de la nouveauté *novelty* dans le profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *Last.fm*

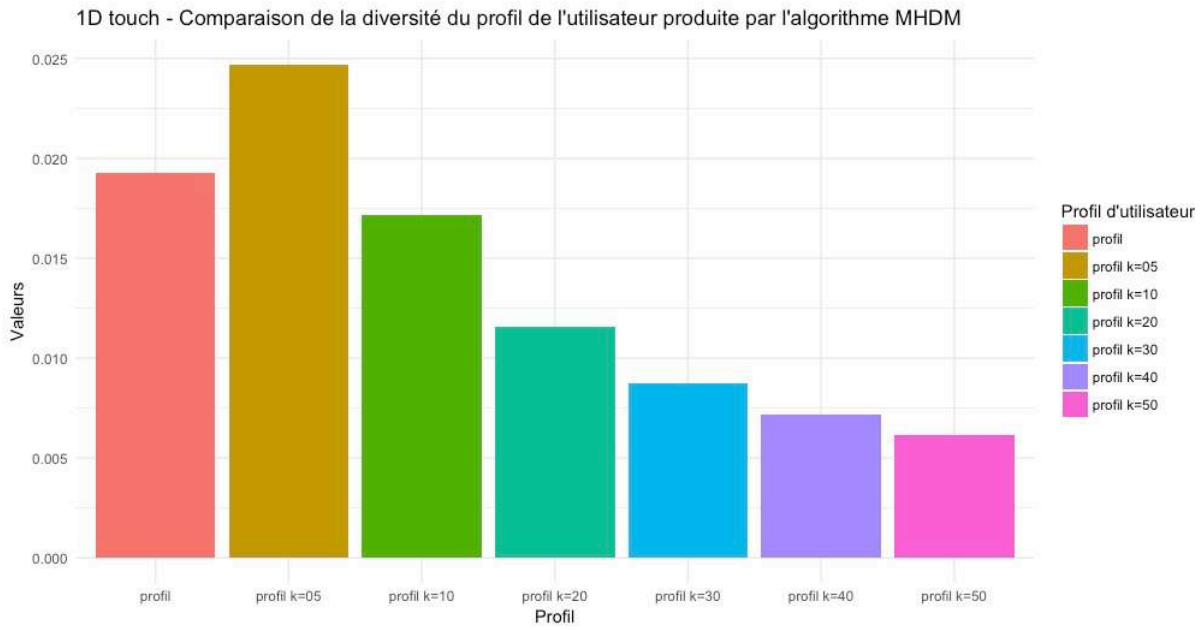



FIGURE 7.11 – Variation de l'*intra-list diversity* du profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *1D touch*

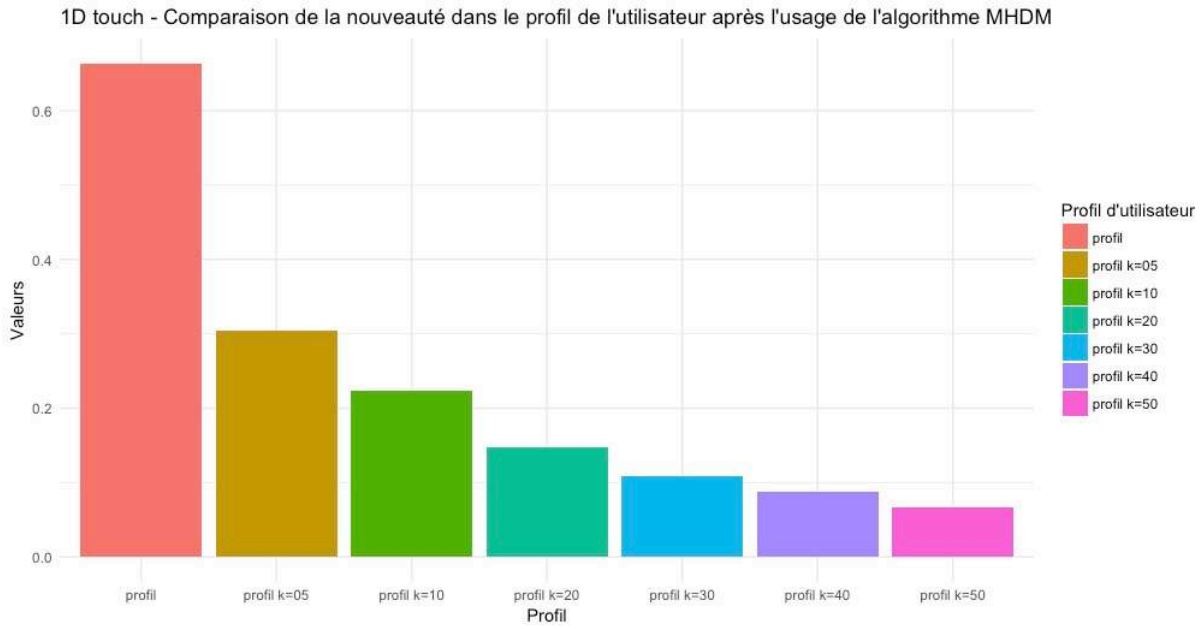



FIGURE 7.12 – Variation de la nouveauté *novelty* dans le profil de l'utilisateur suite à l'application de l'algorithme 4 sur le jeu de données de *1D touch*

#### 7.3.2.2 Comparaison des résultats des algorithmes de reclassement

Dans cette section, nous allons comparer les résultats des modèles de recommandation que nous avons proposés (chapitre 6) aux résultats des modèles de recommandation des algorithmes de reclassement de l'état de l'art (section 7.2.2).

Nous avons décidé de mettre en avant sur chaque jeu de données un type d'algorithme de l'état de l'art pour le reclassement.

Sur le jeu de données de *MovieLens100k*, nous avons sélectionné pour chaque utilisateur sa liste des 100 premières recommandations proposée par l'algorithme *item-based* et nous l'avons reclassée en utilisant les méthodes présentées dans la section 7.2.2 et nos méthodes MPD (section 6.1) et MHDM (section 6.2). Nous avons choisi *item-based* pour étudier le comportement des méthodes de reclassement dans le cas d'un algorithme qui produit des recommandations peu pertinentes (c.-à-d. *précision*, *rappel*, *nDCG*, *Average Precision* faibles), mais qui se comporte mieux sur les métriques de diversification, nouveauté et sérendipité. Nous avons décidé de retenir les résultats avec $\lambda = 0.5$ pour les méthodes MSD, MPD, MMR et IUF. L'usage du paramètre $\lambda = 0.5$ nous permet de faire un bon compromis entre les deux objectifs des fonctions des algorithmes MSD, MMR, MPD et IUF.

La figure 7.13 permet de voir les résultats des algorithmes de reclassement et d'*item-based*. Pour chaque métrique, nous avons pris la moyenne des résultats obtenus pour les listes de longueur $k = [5, 10, 20, 30, 40, 50]$.

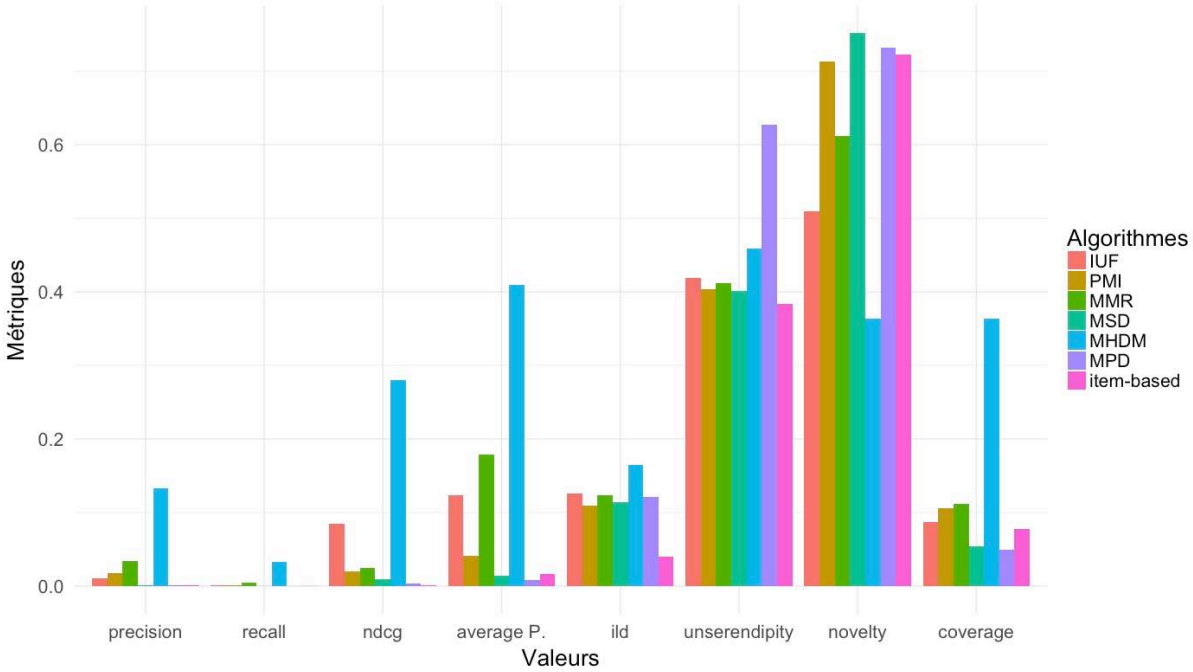



FIGURE 7.13 – Comparaison des résultats des algorithmes de reclassement sur le jeu de données de MovieLens100k

Sur la figure 7.13, nous observons que sur les métriques de précision et de rappel, notre

modèle MHDM basé sur l'ondelette surclasse les autres algorithmes. MMR arrive en deuxième position. Il s'agit ici de deux algorithmes utilisant les informations de contenu pour faire le classement. MSD et notre deuxième modèle MPD basé sur le partitionnement utilisent aussi les informations de contenu, mais obtiennent des résultats en précision et rappel semblables à ceux obtenus par *item-based*. *Item-based* obtient des résultats très faibles dans nos expérimentations sur ce jeu de données. Les algorithmes IUF et PMI se classent en troisième et quatrième position. Ces deux algorithmes utilisent principalement la popularité des articles pour reclasser les articles. On retient un avantage pour IUF. Pour les mesures de classement (nDCG, average Precision), MHDM se détache largement des autres algorithmes. Les algorithmes IUF, MMR, PMI obtiennent aussi de meilleurs résultats qu'*item-based* sur ces deux métriques. MSD arrive à battre *item-based* sur nDCG et obtient à peu près les mêmes résultats qu'*item based* sur la métrique average Precision. Notre second algorithme MPD n'obtient pas de meilleurs résultats qu'*item-based* sur ces métriques. Notre algorithme MHDM surclasse les méthodes de reclassement sur les métriques de pertinence et arrive aussi à mieux classer les résultats.

Pour les mesures qui vont au-delà de la précision, l'écart entre les performances de MHDM et celles des autres algorithmes a considérablement diminué (figure 7.13). Les algorithmes de reclassement arrivent à améliorer les performances d'*item-based* sur certaines métriques. En effet, ils obtiennent de meilleurs résultats en *ild* que ceux obtenus par *item-based* (figure 7.13, cinquième groupe de résultats). Tous les algorithmes arrivent à améliorer l'*ild* d'*item-based*, mais seul l'algorithme de MHDM arrive à se détacher des autres algorithmes. MPD obtient des résultats comparables à ceux obtenus par les autres algorithmes de reclassement.

Le reclassement permet d'obtenir des résultats en *unserendipity* meilleurs que ceux obtenus par *item-based* (figure 7.13). Nos algorithmes MPD et MHDM obtiennent de meilleurs résultats en moyenne que les autres algorithmes sur la métrique *unserendipity*. Nous observons ensuite qu'IUF et MMR arrivent devant PMI et MSD sur cette métrique. Mais les résultats de ces quatre algorithmes sont sensiblement semblables.

Nous avions vu qu'*item-based* obtenait une forte valeur sur la nouveauté. L'algorithme *item-based* arrive à recommander des items peu populaires. Nous observons que le reclassement des listes de résultats par certains algorithmes (IUF, MMR, MHDM) peut faire diminuer considérablement la nouveauté présente dans les listes de recommandation (figure 7.13). Nous remarquons aussi que certains algorithmes (MSD, MPD) arrivent à augmenter le score de nouveauté d'*item-based*. Nous observons que les algorithmes IUF, MMR, MHDM qui ont en moyenne les meilleures valeurs en précision obtiennent en moyenne les moins bonnes valeurs en nouveauté. Notre modèle MHDM n'arrive pas à proposer en moyenne aux utilisateurs les produits se trouvant dans la longue traîne. Notre deuxième modèle MPD arrive en moyenne à proposer un peu plus de nouveauté qu'*item-based*, mais est moins performant que MSD (un des modèles de diversification), le meilleur algorithme de reclassement sur cette métrique sur ce jeu de données. Nous observons pour les autres algorithmes que IUF, qui choisit les produits peu populaires, n'arrive pas à surpasser *item-based*, PMI qui lui aussi recommande des produits peu

populaires, n'arrive à reproduire un score peu ou prou similaire à celui obtenu par *item-based*.

Pour la métrique *coverage*, nous observons que les algorithmes MHDM, MMR, PMI et IUF couvrent plus d'articles du catalogues que *item-based*. Notre modèle MHDM est même le meilleur sur cette métrique. Notre second modèle MPD n'arrive pas à couvrir une grande partie du catalogue et avec MSD il est moins bon qu'*item-based*.

Sur le jeu de données de *Last.fm*, nous avons sélectionné les recommandations d'ALS et nous les avons reclassés en utilisant les méthodes présentées dans la section 7.2.2 et nos méthodes MPD et MHDM présentées au chapitre 6. L'algorithme ALS n'obtient pas les meilleurs scores sur le jeu de données de *Last.fm*, mais est tout de même considéré comme un algorithme phare dans le domaine des recommandations. La figure 7.14 permet de voir les résultats des algorithmes de reclassement sur le jeu de données de *Last.fm*. Ici aussi nous présentons la moyenne des résultats pour $k$ = 5, 10, 20, 30, 40, 50 pour chaque métrique.

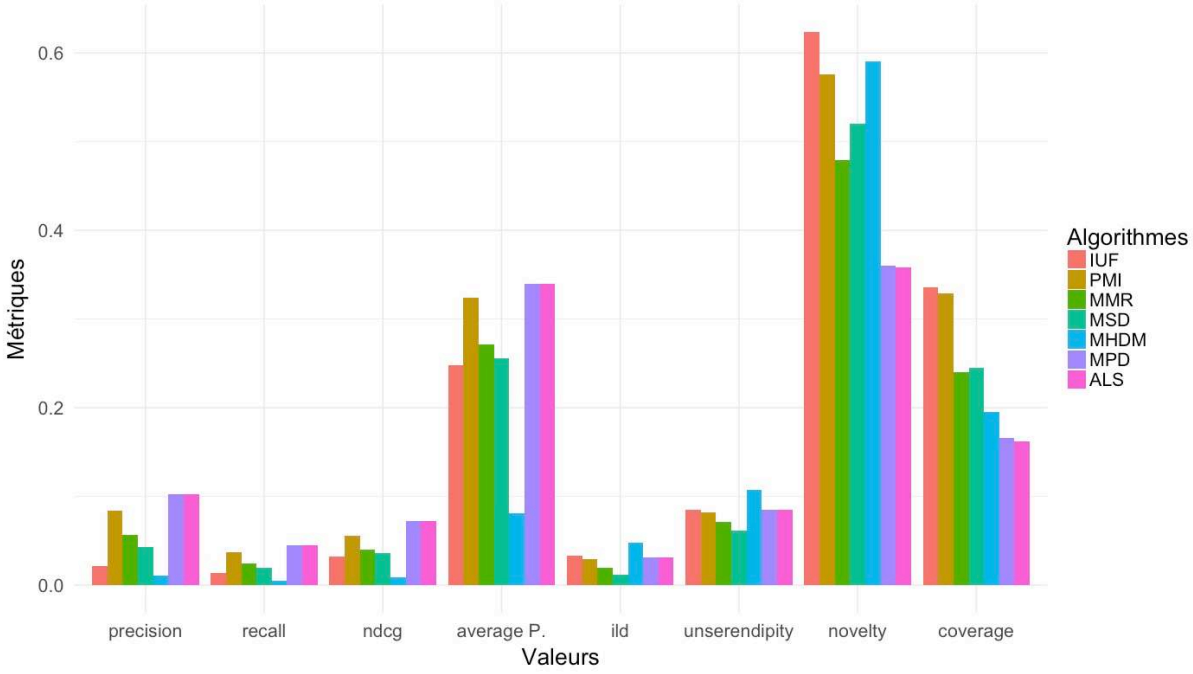



FIGURE 7.14 – Comparaison des résultats des algorithmes de reclassement sur le jeu de données de Last.fm

ALS obtient les meilleurs résultats en précision et rappel. Notre algorithme MPD obtient des résultats semblables à ceux obtenus par ALS. Ensuite on retrouve dans l'ordre PMI, MMR, MSD, IUF, et MHDM. MHDM n'obtient pas les mêmes résultats que sur le jeu de données de *MovieLens100k*. L'écart entre la pertinence de MHDM et des autres algorithmes est même conséquent. Le modèle MHDM contrairement aux autres modèles n'a pas dans sa fonction objectif une sous-fonction pour maximiser la précision. Mais nous observons que la fonction PMI qui ne comporte pas de fonction pour maximiser la précision dans son modèle obtient de meilleurs résultats que MHDM. Ce modèle favorise les couples d'articles qui ont une valeur de cooccurrence très faible dans les ensembles d'entrainement. Ce classement par paires des

résultats permet d'obtenir des couples d'articles peu présents dans le jeu de données mais quand même pertinents.

Sur la diversification, MHDM propose plus de diversité dans ses listes de recommandations que les autres méthodes. Les autres algorithmes à l'exception d'IUF n'améliorent pas la diversité d'ALS. La diversité apportée par MPD est sensiblement égale à celle apportée par ALS. Les algorithmes MMR et MSD qui devraient amener plus de diversité n'obtiennent pas de bons résultats. En effet, nous avons utilisé dans nos expériences les écoutes des utilisateurs pour créer les profils des utilisateurs auxquelles nous avons concaténé les genres et les tags. Or on remarque que l'apport des écoutes des utilisateurs a un effet néfaste sur la diversification des listes de recommandations pour ces deux algorithmes.

Les résultats en *ild* sont semblables aux résultats observés en *unserendipity*. MHDM arrive ici encore à améliorer les résultats d'ALS. À l'exception d'IUF, les autres algorithmes n'améliorent pas l'*unserendipity* d'ALS. Et nous remarquons que MPD obtient un score sensiblement égal à celui obtenu par ALS.

Sur le plan des algorithmes qui ont pour but de proposer des produits nouveaux, IUF et PMI obtiennent les meilleurs résultats parmi dans l'état de l'art. Notre modèle MHDM obtient de moins bons résultats que ceux d'IUF en nouveauté (meilleur algorithme en nouveauté), mais surpasse PMI. MSD et MMR améliorent la nouveauté d'*item-based* et MPD obtient les mêmes valeurs de nouveauté qu'*item-based*. IUF et PMI arrivent à couvrir plus de catalogues que les autres algorithmes. Nos algorithmes n'arrivent pas à toujours battre ceux de l'état de l'art, mais arrivent quand même à améliorer les scores d'*item-based*.

Sur le jeu de données de *1D touch*, nous avons sélectionné les recommandations de BPR et nous les avons reclassées en utilisant les méthodes présentées dans la section 7.2.2 et nos méthodes présentées au chapitre 6. L'algorithme BPR obtient en règle générale les meilleurs résultats de nos expérimentations sur tous les jeux de données.

La figure 7.15 permet de voir les résultats des algorithmes de reclassement sur le jeu de données de *1D touch*. Nous présentons la moyenne des résultats pour $k$ = 5, 10, 20, 30, 40, 50 pour chaque métrique.

Les résultats sur le jeu de données d'1D touch confirment ce qui avait été vu avec ALS : BPR obtient en moyenne les meilleurs résultats. PMI obtient tout de même des résultats en précision comparables aux résultats obtenus par BPR. Les résultats d'IUF et PMI ont toujours le même écart sur les métriques de précision. Notre algorithme MPD obtient des résultats semblables à BPR sur la précision. Sur ce jeu de données, MHDM obtient des résultats en précision semblables à ce qu'il avait obtenu sur *Last.fm*. Notre méthode de reclassement sur des algorithmes tels que BPR et ALS fait considérablement baisser la précision. Sur le rappel, le constat est le même que sur la précision. Les mesures de classement nDCG et Averange rank donnent des résultats semblables. PMI et IUF suivent les mêmes formes de résultats. BPR obtient encore en moyenne les meilleurs résultats sauf pour l' *Averange rank* ou MSD est plus performant.

Sur la diversité, contrairement aux autres jeux de données, MHDM ne produit pas les

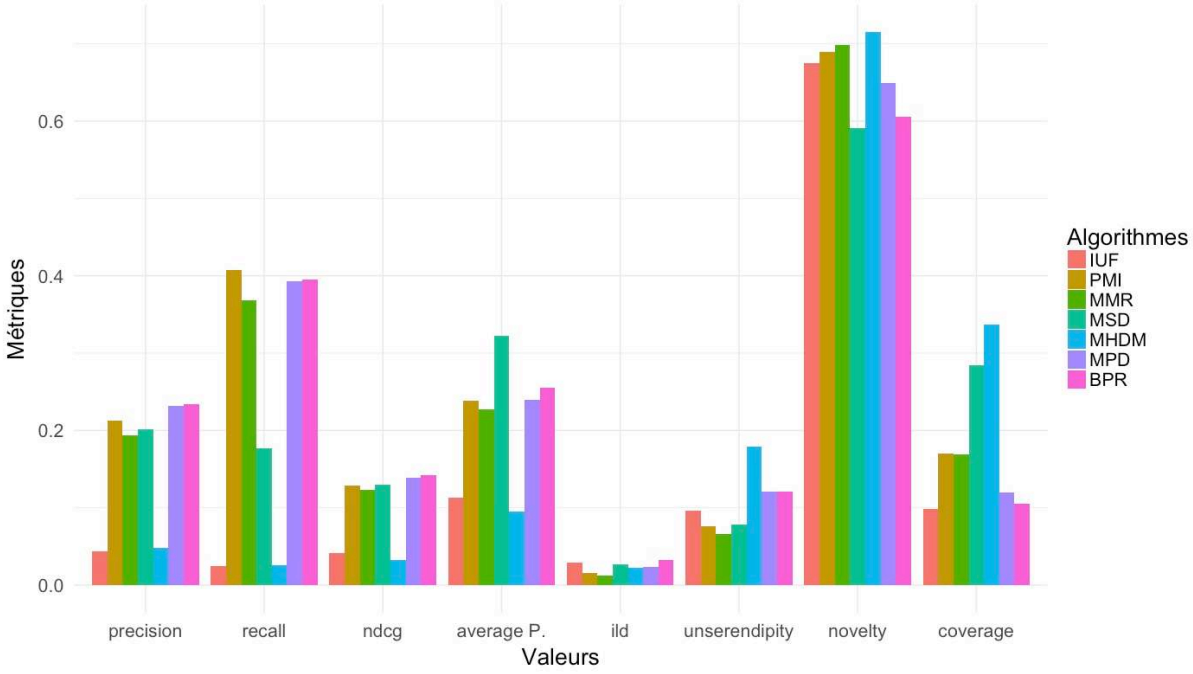



FIGURE 7.15 – Comparaison des résultats des algorithmes de reclassement sur le jeu de données de 1D touch

meilleurs résultats en moyenne. BPR qui obtient une diversité importante reste plus performant sur cette métrique. MPD ne donne pas les mêmes résultats que BPR sur cette métrique. On remarquera aussi que MMR et MSD n'obtiennent toujours pas les meilleurs résultats (pénalisés par le fait que nous avons aussi inséré les écoutes des utilisateurs dans les profils des morceaux).

Sur la sérendipité nous observons des comportements semblables à ce qui avait été relevé sur les autres jeux de données. Notre algorithme MHDM donne encore les meilleurs résultats même si cette fois la baseline BPR est plus forte et obtient de meilleurs résultats que les autres algorithmes de reclassement. Notre second modèle MPD obtient des résultats semblables aux résultats obtenus par BPR.

Pour la nouveauté, notre algorithme MHDM arrive devant les deux meilleurs algorithmes de nouveauté, IUF et PMI. Ces algorithmes ont pour objectif d'augmenter la nouveauté dans les listes de recommandation. Sur la couverture du catalogue d'1D touch, MHDM et MPD arrivent à faire mieux que BPR. Notre modèle MHDM est même le meilleur suivi de MMR. Sur cette métrique, l'algorithme BPR s'est toujours bien comporté dans nos expérimentations.

Nous avons montré les résultats obtenus par les méthodes de reclassement. Nous observons que nos deux modèles MPD et MHDM arrivent à produire des listes de recommandations qui comportent des items nouveaux et divers. Nous remarquons que notre modèle MPD a des résultats semblables aux résultats obtenus par l'algorithme de l'état de l'art. Mais MPD arrive à proposer plus de nouveauté et de diversité que l'algorithme de l'état de l'art notamment sur le jeu de données de *MovieLens*. Nous remarquons d'ailleurs que MPD obtient ses meilleurs résultats sur le jeu de données de *MovieLens* (nous verrons cela dans la section suivante 7.4). MPD est construit

de manière à ce qu'il favorise la diversité (ild) et la nouveauté (*unserendipity*). Notre modèle MHDM a aussi pour mission de favoriser la diversité (ild) et la nouveauté (*unserendipity*). Nous nous rendons compte que notre modèle a de bonnes performances sur ces deux métriques. MHDM arrive aussi à être performant sur la nouveauté (*novelty*) sur le jeu de données d'*1D touch*. Quand MHDM doit reclasser les listes d'algorithmes précis (BPR, ALS), il fait baisser la précision, en revanche quand les algorithmes sont peu précis (item-based), il arrive à proposer des listes précises sans pour autant perdre en diversité ou nouveauté (comme l'indiquent les résultats obtenus sur le jeu de données *MovieLens*). Dans la suite de ce chapitre, nous poursuivons la discussion et nous proposons une conclusion des expérimentations (section 7.4).

## 7.4 Discussion et conclusion

Nous proposons dans cette thèse deux modèles de recommandation qui ont pour but de proposer à des utilisateurs des listes de recommandations qui sont composées d'articles divers et nouveaux. Nos modèles ont aussi la mission d'améliorer la couverture des catalogues. Ces deux modèles ont été présentés dans le chapitre 6 et les résultats ont été présentés dans la section précédente (section 7.3.2.2). Dans les tableaux 7.9, 7.10 et 7.11, nous présentons les résultats à $k$=5 des algorithmes de reclassement de listes de recommandations. Nous avons choisi cette valeur parce qu'il s'agit du nombre de recommandations que nous affichons aux utilisateurs sur la plateforme 1D touch quand ils écoutent un morceau.

| | Item-Based | IUF | PMI | MMR | MSD | MHDM | MPD |
|---|---|---|---|---|---|---|---|
| Précision | 0.0017 | 0.0271 | 0.0098 | *0.1049* | 0.0017 | **0.1254** | 0.0017 |
| Rappel | 1.15e-05 | 0.0011 | 0.0002 | *0.0065* | 4.97e-05 | **0.0086** | 1.15e-05 |
| nDCG | 0.0014 | *0.0849* | 0.0238 | 0.0566 | 0.0089 | **0.1743** | 0.0029 |
| Average P. | 0.0043 | 0.1056 | 0.0368 | *0.2821* | 0.0089 | **0.2945** | 0.0046 |
| ILD | 0.0395 | 0.1321 | 0.1189 | *0.1325* | 0.1091 | **0.1615** | 0.1210 |
| Unserendipity | 0.7464 | **0.9144** | 0.9135 | *0.9142* | 0.9127 | 0.7940 | 0.9136 |
| Novelty | *0.7395* | 0.3546 | 0.7331 | 0.3882 | **0.7499** | 0.2549 | *0.7395* |
| Coverage | 0.0291 | 0.0351 | *0.0456* | **0.1364** | 0.0159 | 0.0434 | 0.0291 |

Tableau 7.9 – Résultats expérimentaux des algorithmes de reclassement sur le jeu de données de MovieLens avec des listes de recommandations de longueur $k = 5$.

| | ALS | IUF | PMI | MMR | MSD | MHDM | MPD |
|---|---|---|---|---|---|---|---|
| Précision | **0.1466** | 0.0132 | *0.1258* | 0.0848 | 0.0616 | 0.0144 | **0.1466** |
| Rappel | **0.0150** | 0.0013 | *0.0129* | 0.0085 | 0.0062 | 0.0014 | **0.0150** |
| nDCG | **0.1416** | 0.0237 | *0.0866* | 0.0729 | 0.0449 | 0.0174 | **0.1416** |
| Average P. | **0.2949** | 0.0436 | *0.2458* | 0.1857 | 0.1362 | 0.0426 | **0.2949** |
| ILD | 0.0311 | *0.0328* | 0.0273 | 0.0205 | 0.0072 | **0.0403** | 0.0309 |
| Unserendipity | 0.2386 | *0.2497* | 0.2258 | 0.2028 | 0.1763 | **0.2778** | 0.2386 |
| Novelty | 0.3245 | **0.7178** | *0.6372* | 0.4219 | 0.4901 | 0.6029 | 0.3259 |
| Coverage | 0.0724 | *0.2177* | **0.2220** | 0.1036 | 0.1166 | 0.0393 | 0.0739 |

Tableau 7.10 – Résultats expérimentaux des algorithmes de reclassement sur le jeu de données de last.fm avec des listes de recommandations de longueur $k = 5$.

Notre premier modèle, MPD, se base sur un clustering des préférences de l'utilisateur pour produire des listes de recommandations avec des éléments différents les uns des autres. Le modèle est bâti de sorte que la diversité soit personnalisée. Sur les différents jeux de données notre proposition MPD obtient des résultats semblables à ceux obtenus par les algorithmes de l'état de l'art pour les métriques de précision et de rappel. Quand l'algorithme de l'état de l'art utilisé obtient de forts résultats en précision et rappel, MPD obtient aussi de forts résultats, par exemple MPD a un score de 0.3799 pour la précision à $k = 5$ et BPR a un score de 0.3767 à $k = 5$ toujours sur la précision. BPR obtient sur toutes les expériences les meilleurs résultats en précision et en rappel parmi les algorithmes de l'état de l'art que nous avons choisi. Quand l'algorithme de l'état

| | BPR | IUF | PMI | MMR | MSD | MHDM | MPD |
|---|---|---|---|---|---|---|---|
| Précision | *0.3799* | 0.1096 | **0.5770** | 0.3473 | 0.3091 | 0.0884 | 0.3767 |
| Rappel | 0.2614 | 0.0163 | **0.3846** | *0.3037* | 0.2683 | 0.0227 | 0.2818 |
| nDCG | *0.2965* | 0.1087 | **0.4351** | 0.2508 | 0.2385 | 0.0798 | 0.2851 |
| Average P. | *0.3431* | 0.1234 | **0.3768** | 0.3395 | 0.2764 | 0.1479 | 0.3285 |
| ILD | *0.0295* | 0.0087 | 0.0178 | 0.0166 | 0.0068 | **0.0300** | 0.0194 |
| Unserendipity | 0.3518 | 0.1881 | 0.2062 | 0.2231 | 0.1756 | **0.5578** | *0.3499* |
| Novelty | 0.5807 | **0.7867** | 0.6602 | 0.6766 | *0.6978* | 0.6876 | 0.6316 |
| Coverage | 0.0462 | 0.0709 | *0.1132* | 0.0975 | 0.0986 | **0.1160** | 0.0647 |

Tableau 7.11 – Résultats expérimentaux des algorithmes de reclassement sur le jeu de données d'1D touch avec des listes de recommandations de longueur $k = 5$.

de l'art utilisé obtient de faibles résultats en précision et rappel, MPD obtient aussi de faibles résultats, par exemple MPD a un score de 0.0017 pour la précision à $k = 5$ et *item-based* a un score de 0.0017 à $k = 5$. L'algorithme MPD que nous avons mis en place quand il n'arrive pas à créer de clusters pour un utilisateur, applique la méthode MSD et quand cette méthode ne peut être appliquée parce que l'ensemble d'entrainement de l'utilisateur est vide, nous gardons la liste de recommandations retrouvée par l'algorithme de l'état de l'art. L'algorithme MPD que nous avons proposé fonctionne bien quand le profil de l'utilisateur contient plusieurs articles. Sur le jeu de données de *MovieLens1k* MPD obtient de bons résultats (tableaux 7.9). Il arrive à garder une précision et un rappel proche de ce que l'algorithme de l'état de l'art obtient, tout en améliorant l'*ild* et l'*unserendipity*.

L'amélioration de l'ild et de l'unserendipity sont possibles grâce au clustering. Le clustering est intéressant sur le jeu de données de *MovieLens1k*, car la diversité dans le profil des utilisateurs de ce jeu de données est plus grande que celle des autres jeux de données. Le clustering assure l'existence d'une diversité dans la liste de recommandation. MPD sélectionne ensuite pour chaque cluster un article. De plus dans la formule d'objectif du modèle MPD ($D(i) = \arg\max_{c \in C}(sim(i,c)) \cdot \frac{1}{|S|} \sum_{j \in S} (dist(i,j))$), nous nous assurons aussi que l'article choisi est différent des articles déjà dans la liste de recommandations. Le clustering comme nous l'utilisons permet aussi d'améliorer la sérendipité. La sérendipité que nous calculons est la différence entre les produits dans la liste de recommandation et ceux qui sont dans le profil. Le modèle MPD prend en compte l'éloignement par rapport au profil de l'utilisateur à partir de la similarité entre l'article et le vecteur centroid du cluster ($sim(i,c)$). MPD choisit l'article qui obtient la valeur maximale du produit $(sim(i,c)) \cdot \frac{1}{|S|} \sum_{j \in S} (dist(i,j))$, soit l'article le plus similaire du profil de l'utilisateur et aussi le plus différent des autres articles déjà sélectionnés. Les résultats de MPD sont meilleurs que ceux obtenus par MSD sur le jeu de données *MovieLens1k*. Sur les autres jeux de données l'apport de MPD n'est pas aussi visible que sur celui de *MovieLens1k* (tableaux 7.10, 7.11). Le jeu de données de *MovieLens* est régulier, en effet presque tous les utilisateurs ont regardé 57 films. Les autres jeux de données surtout celui d'1D touch ne sont pas favorables au déploiement d'une telle méthode car les profils d'écoute des utilisateurs d'1D touch contiennent peu de

morceaux (tableau 7.8).

| | MPD-SVD | MPD | Cluster |
|---|---|---|---|
| Précision | 0.0017 | 0.0017 | 0.001 |
| Rappel | 1.15e-05 | 1.15e-05 | 1.13e-05 |
| nDCG | 0.0023 | 0.0029 | 0.0009 |
| Average P. | 0.004 | 0.0046 | 0.0002 |
| ILD | 0.01184 | *0.1210* | **0.1310** |
| Unserendipity | 0.7464 | *0.9136* | 0.6284 |
| Novelty | *0.7395* | *0.7395* | **0.8135** |
| Coverage | 0.0291 | 0.0291 | 0.03O2 |

Tableau 7.12 – Résultats expérimentaux de l'algorithme de reclassement MPD sur le jeu de données de MovieLens avec des listes de recommandations de longueur $k = 5$.

Nous présentons dans le tableau 7.12, les résultats des différentes méthodes de MPD. En effet, la version de MPD que nous avons présenté jusqu'à présent est celle qui utilise la somme des vecteurs des articles pour construire un profil pour l'utilisateur. Mais nous pouvons aussi modéliser les profils des utilisateurs en faisant une factorisation de matrice (chapitre 6, section 6.1 : modélisation des utilisateurs). Dans le tableau 7.12, nous présentons aussi les résultats de la méthode basée sur un clustering des préférences des utilisateurs (chapitre 6, section 6.1 : méthode basée sur la similarité : Clustering des préférences des utilisateurs). Nous appelons cette méthode *Cluster* dans le tableau 7.12. Nous n'avons reporté que les résultats pour le jeu de données *MovieLens1k* à $k = 5$. Pour la version SVD de MPD, les résultats sont sensiblement semblables à ceux de MPD (version présentée) sauf sur la métrique ild, où la version présentée obtient de meilleurs résultats. La version *Cluster*, inspirée des systèmes de recommandation basés sur le contenu arrive grâce au clustering à obtenir un bon score en *ild*. Le clustering permet de répondre au problème de liste de recommandations contenant des articles trop similaires entre eux. C'est un problème inhérent aux systèmes de recommandations basés sur le contenu. La version *Cluster* propose même plus de nouveauté que MPD et réalise aussi une meilleure couverture du catalogue. Mais comme pour un système basé sur le contenu, cette version basée sur le clustering produit des listes proches du profil de l'utilisateur, d'où l'intérêt de la méthode MPD et de la formule $D(i) = \arg\max_{c \in C}(sim(i, c)) \cdot \frac{1}{|S|} \sum_{j \in S} (dist(i, j))$. Cependant l'approche MPD dépend de la taille des profils des utilisateurs. Elle dépend aussi de la diversité du profil de l'utilisateur. Si ces conditions sont réunies, MPD pourra faire un clustering des préférences de l'utilisateur, et améliorera la diversité dans la liste de recommandations.

Notre deuxième modèle, MHDM, est pensé pour répondre surtout au problème de personnalisation accrue. Cette méthode diversifie les listes de recommandations, mais n'applique pas une méthode gloutonne comme MSD et MMR (les deux méthodes de diversification). Notre modèle MHDM comme MPD recherche la diversification dans le profil de l'utilisateur. Contrairement au modèle MPD, le modèle MHDM ne fait pas un clustering des préférences de l'utilisateur. Il recherche les articles les plus singuliers du profil de l'utilisateur afin de créer des listes de

recommandations diverses à partir des articles sélectionnés.

MHDM a deux comportements distincts : pour le reclassement des recommandations d'un algorithme qui obtient de faibles résultats sur les métriques de précision, MHDM améliore les résultats sur ces métriques, 0.1254 en précision pour MDHM et 0.0017 en précision pour *item-based*, pour le reclassement des recommandations d'un algorithme plus performant sur les métriques de précision, MHDM a tendance à dégrader les résultats en précision, 0.0884 pour MHDM et 0.3799 pour BPR. Notre méthode MHDM comme les autres méthodes de reclassement fait remonter en tête de liste les articles pertinents qu'un algorithme comme *item-based* place malencontreusement en queue de liste (tableau 7.9). Mais cette faculté que possèdent les méthodes comme MHDM d'être beaucoup plus précis que l'algorithme de l'état de l'art fait diminuer l'apport en nouveauté dans les listes de recommandations. La nouveauté ici est le pourcentage d'articles de la longue traîne qui se retrouvent dans les listes de recommandations. Mais MHDM réussi à recommander des articles plus différents les un des autres. Dans les tableaux 7.9, 7.10, 7.11, l'ild de MHDM est supérieur à l'ild des autres algorithmes. Nous avons utilisé un paramètre $\lambda = 0.5$ sur les autres méthodes de diversification (MMR et MSD). Ce paramètre est utilisé sur ces méthodes pour faire le compromis entre la diversité et la précision. Nous avons choisi ce paramètre parce qu'il n'avantage aucune des métriques (précision, ild). Mais en utilisant un paramètre $\lambda$ qui favorise la diversité, MHDM se fait battre par MMR (par exemple, ild= 0.0503 sur *MovieLens* à $k$= 5) et MSD (par exemple, ild= 0.0442 sur *MovieLens* à $k$= 5).

Contrairement aux autres méthodes MMR et MSD, MHDM ne fait pas une sélection en maximisant la diversité entre les articles sélectionnés à partir d'un algorithme glouton. L'ild est assurée par l'algorithme 4 (section 6.2), et dans l'une des sections précédentes (section 7.3.2.1) nous avons montré que cet algorithme est performant quand les profils des utilisateurs contiennent un nombre suffisant d'articles. Pour les listes de recommandation comportant peu d'items ($k$ =5 et $k$ = 10), l'usage de cet algorithme est pertinent. Mais pour des utilisateurs qui n'ont pas eu beaucoup d'interactions avec le système, cet algorithme présente peu d'intérêt (par exemple sur le jeu de données d'*1D touch*). Pour pallier ce problème quand le nombre d'articles présents dans le profil de l'utilisateur est inférieur au nombre d'articles à sélectionner par l'algorithme 4, MHDM fait la somme des vecteurs des articles pour créer le profil de l'utilisateur (comme dans le chapitre 6, section 6.1 : modélisation des utilisateurs par la somme des vecteurs des articles). Cette méthode est utilisée sur le jeu de données d'*1D touch* pour les k > 5, avec peu de succès puisque l'ild diminue et est inférieure à celle de l'algorithme de l'état de l'art et des autres méthodes.

En revanche, en utilisant la formule du Chapeau mexicain, le modèle MHDM sélectionne des articles qui sont différents de ceux que l'utilisateur a appréciés dans le passé. Le Chapeau mexicain se base sur la distribution des distances entre l'article qui sert de source (celui qui est sélectionné par l'algorithme 4) et les autres articles candidats à la recommandation pour sélectionner un article se trouvant dans une zone intermédiaire, entre le trop proche de l'article

source et le trop éloigné de l'article source. Les résultats en *Unserendipity* attestent de la pertinence du modèle. Ce dernier propose des articles qui sont éloignés du profil de l'utilisateur. Quand MHDM est utilisé pour reclasser un algorithme de recommandation précis, comme BPR, il arrive à obtenir des meilleurs résultats en *Unserendipity* que les autres algorithmes. MHDM va chercher les articles qui ont été classés en queue de liste par BPR pour les mettre en tête de liste. Ces articles sont différents de ceux que l'utilisateur avait déjà écoutés. Pour l'algorithme BPR, cela améliore aussi la nouveauté dans les listes de recommandation, c'est-à-dire les articles qui sont dans la longue traîne. Mais MHDM n'arrive pas toujours à battre les modèles bâtis pour retrouver les articles qui sont dans la longue traîne tel que IUF et PMI (tableaux 7.10, 7.11). La formule utilisée dans le modèle MDHM contrairement à IUF et PMI ne se base pas sur le nombre d'interactions ou la probabilité d'interactions pour recommander les articles. Le modèle MHDM recommande des articles éloignés du profil de l'utilisateur en se basant sur le contenu. Les systèmes qui se basent sur le contenu arrivent cependant à recommander des contenus de longue traîne (section 7.3.1.1. L'ADN des algorithmes basés sur le contenu se retrouve dans MHDM au niveau de la couverture du catalogue. Sur les listes de longueur $k$= 5, MHDM n'est pas toujours leader sur la métrique de *coverage*, mais quand le nombre de recommandations augmente, MHDM offre une meilleure couverture du catalogue que les autres modèles.

Les deux modèles (MPD et MHDM) que nous présentons cherchent à offrir aux utilisateurs des listes de recommandations qui contiennent des élèments différents entre eux, et aussi différents de ce que l'utilisateur a déjà apprécié dans le passé. MPD est un algorithme qui obtient des résultats similaires aux résultats que l'algorithme de l'état de l'art obtient. Néanmoins, quand le jeu de données comporte des profils d'utilisateurs qui sont fournis en item, MPD augmente la diversité des listes de recommandation. MHDM réussit lui un meilleur travail de ce côté tout en arrivant aussi à proposer aux utilisateurs des contenus différents de ce qu'ils ont déjà expérimentés. Ces deux modèles ne sont pas des spécialistes pour remonter des éléments de la longue traîne, mais offrent de bons résultats pour obtenir des performances en adéquation avec ceux obtenus par d'autres modèles spécialement conçus pour cette tâche.

Troisième partie

# Conclusion générale

Chapitre 8

# Un besoin de diversité et de nouveauté dans le monde

Cette thèse s'est articulée autour du problème de diversité et de nouveauté dans les systèmes de recommandation. La grande masse de données accessible sur les plateformes web a poussé à mettre en place des prescripteurs automatiques. Ces prescripteurs recommandent aux utilisateurs des produits sans même attendre que ces derniers expriment un besoin explicite. En se basant sur les comportements des utilisateurs, le système de recommandation agit de manière indépendante et fournit à l'utilisateur des produits qui pourraient l'intéresser. L'engouement pour les systèmes de recommandation et la volonté de toujours proposer aux individus des produits pertinents a pénalisé d'autres aspects à prendre en compte dans le processus de consommation des individus. L'envie de découverte, de nouveauté, de diversité sont autant d'aspects à prendre en compte lors de la création de listes de recommandations. Contrairement à un système de recherche d'information où l'utilisateur indique clairement sa demande d'information, le système de recommandation peut se permettre d'être moins centré sur les envies des utilisateurs en leur proposant d'explorer des produits qui s'éloignent de leurs centres d'intérêt immédiats, surtout pour les données culturelles. Les listes de recommandations ne peuvent pas être seulement une sélection de produits qui reflètent les comportements passés des utilisateurs. Nous pensons que nous pouvons ajouter de l'« audace » dans ces listes en sortant des sentiers battus [Schwartz, 2004]. En revanche nous ne pouvons pas donner trop de diversité ou de nouveauté dans nos listes de recommandations, sous peine de nous éloigner des intérêts des utilisateurs.

Nous pouvons tirer un parallèle avec les travaux des psychologues spécialisés dans le développement de l'enfant comme ceux de Piaget [1923], Winnicott [1975] et Stern [1989]. Un petit enfant à besoin de routine, d'un rythme qui se répète pour trouver des constantes. Cela lui donne un sentiment de sécurité sur lequel il va construire son identité. Néanmoins il a également besoin d'expériences nouvelles, de diversité afin de construire son imaginaire et développer son intelligence [Schneider et McGrew, 2012]. Néanmoins, cette nouveauté doit être adaptée à son rythme propre pour qu'il assimile ces nouvelles informations.

La recommandation peut donc être un modèle adapté à chacun et qui permet aux gens de prendre à petite dose des informations nouvelles et diverses. Les travaux de recherche présentés dans ce document ont porté sur l'usage de techniques en fouille de données pour proposer de la diversité et de la nouveauté dans les systèmes de recommandation tout en maintenant un ancrage sur les habitudes des utilisateurs. Nous avons proposé deux modèles qui apportent de la

nouveauté et de la diversité dans les listes de recommandations. Le premier modèle, *Maximal Personalized Diversity* (MPD), a montré qu'il pouvait être une alternative aux autres modèles de diversification pure des systèmes de recommandation. Ce modèle maintient des taux de diversification comparables aux autres modèles et améliore la pertinence des recommandations. Ceci s'explique par l'apport d'une diversité personnalisée en se basant sur un clustering des préférences de l'utilisateur. Le second modèle, *Mexican-Hat Diversity Model* (MHDM), se base sur la dérivée seconde de la loi normale pour sélectionner selon leur contenu les produits à recommander. Ce modèle ne cherche pas à maximiser la diversité dans les listes de recommandation, mais arrive grâce à ses propriétés à sélectionner des articles nouveaux pour l'utilisateur sans pour autant trop s'éloigner des préférences de l'utilisateur. Les éléments essentiels à retenir de ces deux modèles présentés dans ce travail sont les suivants :

1. il s'agit de systèmes de recommandation qui cherchent à promouvoir la nouveauté et la diversité comme vertu pédagogique de manière à couvrir le spectre d'intérêt de l'utilisateur et aussi à l'élargir. Le système ne tombera pas dans la recommandation « facile » des produits populaires qui vont plaire à tout le monde, mais qui peuvent frustrer sur le long terme. Inversement, le système ne va pas effectuer trop de personnalisation sur les articles déjà appréciés par l'utilisateur. Une recommandation basée sur la seule personnalisation effectuée à travers les articles déjà appréciés pourrait induire la répétition infinie des comportements de l'utilisateur ou de l'utilisatrice du système produisant une version étroite, bornée de lui-même ou d'elle-même Pariser [2011] ;
2. il s'agit aussi de systèmes de recommandations qui pour les sites de *streaming* pourront mieux couvrir le catalogue, atteignant les articles de la longue traîne comme cela était attendu avec la révolution numérique [Anderson, 2008] ;
3. il s'agit enfin de systèmes de recommandation qui auront tendance à recommander l'ensemble des créateurs de contenu, et ceci de manière équitable, sans privilégier les plus populaires, donnant à chacun l'opportunité de contribuer à la diversité culturelle du monde.

Dans un monde où l'intelligence artificielle va continuer à jouer un rôle prépondérant entre nous et les produits et services en nous accompagnant dans nos choix, il nous semble que nos modèles sont des réponses aussi originales que durables au questionnement de la diversité et de la nouveauté dans le monde.

## Le marché de la donnée, le marché de la confiance

À l'heure où j'écris ces lignes, le rapport de Cédric Villani [Villani, 2018] a été remis au gouvernement français après 6 mois de travail. La première proposition de la 5$^{ème}$ partie de ce rapport (« Quelle éthique de l'IA ?) » concerne l'accroissement de la transparence et l'auditabilité des systèmes autonomes, autrement dit « ouvrir la boîte noire ». La principale ligne de ce rapport est de mettre en avant une IA responsable et soucieuse des utilisateurs. La « bulle de filtres »,

que nous avons souhaité faire éclater dans les travaux de recommandation réalisés durant cette thèse, nous semble justement résulter du manque d'explicabilité des algorithmes classiques de recommandation et de personnalisation : les utilisateurs de ces systèmes ne savent pas que les informations extérieures qui ne sont pas parfaitement conformes à leurs attentes finissent par ne plus leur parvenir, les enfermant dans une forme de communautarisme digital.

## Une IA au service de la société

La partie suivante de ce rapport (« Pour une intelligence artificielle inclusive et diverse ») s'ouvre sur le paragraphe suivant :

> L'intelligence artificielle ne peut pas être une nouvelle machine à exclure. C'est une exigence démocratique dans un contexte où ces technologies sont en passe de devenir une des clés du monde à venir. Elle ouvre de forminables opportunités de création de valeur et de développement de nos sociétés et des individus. Ces opportunités doivent bénéficiés à tous.

Rappelons que cette thèse s'est déroulée à la fois dans un laboratoire de recherche et au sein d'une entreprise de l'économie sociale et solidaire. Nos travaux ont été réalisés avec le souci de pouvoir s'appliquer à **tous**, et en particulier à la création indépendante, comme la « musique indé », c'est-à-dire celle qui n'est pas soutenue par les gros labels de l'industrie musicale. Avec l'arrivée de la révolution numérique, en effet, les artistes (musiciens, auteurs, développeurs de jeux vidéo...) qui ne sont pas rattachés à des grosses structures sont moins visibles et leurs créations ont plus de peine à trouver un public. À travers nos algorithmes de recommandation audacieux, non seulement nous cherchons à renouveler l'univers culturel du public en dépassant la frontière de la bulle de filtre, mais en plus nous essayons de favoriser la connaissance des travaux des créateurs indépendants par le public.

CHAPITRE 9

# PERSPECTIVES

Les systèmes de recommandations sont de bons outils pour amener de la nouveauté ou de la diversité aux utilisateurs sur les sites en ligne. Mais comme nous l'avons montré dans cette thèse, les systèmes de recommandation dépendent avant tout des données disponibles. Nous sommes conscients que des données enrichies et structurées amélioreront les performances des systèmes de recommandation et permettront de trouver de nouvelles façons de proposer des recommandations.

Or, les artistes indépendants, émergents, les artisans de la création, souffrent d'un manque de visibilité sur les réseaux. Ce phénomène s'explique par un manque de qualification et de documentation sur leurs données ou tout simplement une absence de données (contrairement aux artistes *mainstream*). L'une des missions d'1D Lab est de faire découvrir et émerger ces artistes en augmentant leur capacité à rencontrer leurs publics. Mais cette carence de données rend difficile toute forme de recommandations automatiques basées sur le contenu et enlève de fait ces artistes des résultats de recherche.

## 9.1 Discover it

1D Lab travaille ainsi à proposer de nouveaux chemins de découverte, augmenter l'appétence musicale des usagers, simuler la sérendipité. L'enjeu est notamment de structurer une base de connaissances décrivant le monde de la culture indépendante. Il s'agira dans un premier temps de regrouper les données sur ces univers et ensuite les transformer dans un format exploitable par les ordinateurs. La modélisation des connaissances permettra d'identifier des concepts, de les classer, et surtout de raisonner sur le domaine. Le processus de création de la base de connaissances devra comprendre une étape de modélisation sémantique des données. Cette étape est primordiale, parce qu'elle définit le format et la structure des données, elles constituent une véritable ontologie du domaine. Une ontologie est la définition formelle de termes dans un domaine et les relations qui existent entre eux [Gruber, 1993]. Nous pourrons en partie composer à partir d'ontologies existantes comme la *Music Ontology* [1]. Il faudra initialiser cette base de connaissances à partir de données publiques, ouvertes sur le web, ou d'autres sources du web des données bien identifiées

1. `http://musicontology.com/`

(*DBpedia* [2], *MusicBrainz* [3], *Freebase* [4]). Pour finir, on devra passer par une étape de consolidation des données. Nous avons indiqué que les artistes indépendants souffraient d'un manque de données et parfois ces données venaient à manquer. Pour résoudre ce problème, nous voulons mettre à disposition un outil collaboratif qui permettra à des « experts », des acteurs du projet, mais également à la communauté des professionnels, à des artistes et à des amateurs « éclairés », un moyen mutualisé de compléter, de corriger et de créer de l'information dans la base.

En résumé, cette base de connaissances entend partager une connaissance structurée commune de la musique et de manière générale la culture indépendante avec des personnes et des ordinateurs [Musen, 1992; Gruber, 1993]. Elle pourra de plus être réutilisable. Si nous voulons créer à l'avenir une ontologie qui couvrirait un domaine plus large que la culture indépendante, nous pourrions utiliser des parties de notre ontologie ou d'autres ontologies qui décrivent le domaine à définir. La base de connaissances nous permettra d'améliorer encore les performances de nos algorithmes de découvertes de contenus culturels indépendants. Sa mise en place est un processus créatif. Notre vision et notre compréhension de la culture indépendante vont fortement influencer sa structure. Deux ontologies définies par des personnes différentes ne seront jamais semblables. C'est grâce à la diversité des connaissances, des compétences et des idées qui se trouvent dans la collectivité que pourra se constituer une ontologie aux sensibilités plurielles, la mieux à même, selon nous, de servir un processus de recommandation cherchant à mettre en avant les facteurs de nouveauté et de diversité dont le monde a besoin.

2. `http://fr.dbpedia.org/`
3. `https://musicbrainz.org/`
4. `https://www.freebase.com/`

# BIBLIOGRAPHIE


Zeinab ABBASSI, Sihem AMER-YAHIA, Laks V. S. LAKSHMANAN, Sergei VASSILVITSKII et Cong YU : Getting recommender systems to think outside the box. *In* Lawrence D. BERGMAN, Alexander TUZHILIN, Robin D. BURKE, Alexander FELFERNIG et Lars SCHMIDT-THIEME, éditeurs : *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009*, pages 285–288. ACM, 2009. URL http://doi.acm.org/10.1145/1639714.1639769.

Panagiotis ADAMOPOULOS et Alexander TUZHILIN : On unexpectedness in recommender systems : Or how to better expect the unexpected. *ACM TIST*, 5(4):54 :1–54 :32, 2014. URL http://doi.acm.org/10.1145/2559952.

Gediminas ADOMAVICIUS et Youngok KWON : Towards more diverse recommendations : Item re-ranking methods for recommender systems. *In In Workshop on Information Technologies and Systems*, 2009.

Gediminas ADOMAVICIUS et YoungOk KWON : Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. Knowl. Data Eng.*, 24(5):896–911, 2012. URL https://doi.org/10.1109/TKDE.2011.15.

Gediminas ADOMAVICIUS et YoungOk KWON : Optimization-based approaches for maximizing aggregate recommendation diversity. *INFORMS Journal on Computing*, 26(2):351–369, 2014. URL https://doi.org/10.1287/ijoc.2013.0570.

Gediminas ADOMAVICIUS et Alexander TUZHILIN : Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749, 2005. URL https://doi.org/10.1109/TKDE.2005.99.

Gediminas ADOMAVICIUS et Alexander TUZHILIN : Context-aware recommender systems. *In* Francesco RICCI, Lior ROKACH et Bracha SHAPIRA, éditeurs : *Recommender Systems Handbook*, pages 191–226. Springer, 2015. URL https://doi.org/10.1007/978-1-4899-7637-6_6.

Rakesh AGRAWAL, Sreenivas GOLLAPUDI, Alan HALVERSON et Samuel IEONG : Diversifying search results. *In* Ricardo A. BAEZA-YATES, Paolo BOLDI, Berthier A. RIBEIRO-NETO et Berkant Barla CAMBAZOGLU, éditeurs : *Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009*, pages 5–14. ACM, 2009. URL http://doi.acm.org/10.1145/1498759.1498766.

Takayuki Akiyama, Kiyohiro Obara et Masaaki Tanizaki : Proposal and evaluation of serendipitous recommendation method using general unexpectedness. *In* Jérôme Picault, Dimitre Kostadinov, Pablo Castells et Alejandro Jaimes, éditeurs : *Proceedings of the Workshop on the Practical Use of Recommender Systems, Algorithms and Technologies, PRSAT 2010, Barcelona, Spain, September 30, 2010.*, volume 676 de *CEUR Workshop Proceedings*, pages 3–10. CEUR-WS.org, 2010. URL http://ceur-ws.org/Vol-676/paper1.pdf.

Pek van Andel : Anatomy of the unsought finding. Serendipity : Origin, history, domains, traditions, appearances, patterns and programmability. *The British Journal for the Philosophy of Science*, 45(2):631–648, 1994.

Chris Anderson : *The Long Tail : Why the Future of Business Is Selling Less of More*. Hachette Books, New York, New York, USA, 2008.

Azin Ashkan, Branislav Kveton, Shlomo Berkovsky et Zheng Wen : Optimal greedy diversity for recommendation. *In* Qiang Yang et Michael Wooldridge, éditeurs : *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1742–1748. AAAI Press, 2015. URL http://ijcai.org/Abstract/15/248.

Ricardo A. Baeza-Yates et Berthier A. Ribeiro-Neto : *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999. URL http://www.dcc.ufmg.br/irbook/.

Eytan Bakshy, Solomon Messing et Lada A. Adamic : Exposure to ideologically diverse news and opinion on facebook. 348(6239):1130–1132.

Trapit Bansal, David Belanger et Andrew McCallum : *Ask the GRU* : Multi-task learning for deep text recommendations. *In* Shilad Sen, Werner Geyer, Jill Freyne et Pablo Castells, éditeurs : *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 107–114. ACM, 2016. URL http://doi.acm.org/10.1145/2959100.2959180.

Oren Barkan et Noam Koenigstein : ITEM2VEC : neural item embedding for collaborative filtering. *In* Francesco A. N. Palmieri, Aurelio Uncini, Kostas I. Diamantaras et Jan Larsen, éditeurs : *26th IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2016, Vietri sul Mare, Salerno, Italy, September 13-16, 2016*, pages 1–6. IEEE, 2016. URL https://doi.org/10.1109/MLSP.2016.7738886.

Pierpaolo Basile, Marco De Gemmis, Anna Lisa Gentile, Pasquale Lops et Giovanni Semeraro : UNIBA : JIGSAW algorithm for word sense disambiguation. *In* Eneko Agirre, Lluís Màrquez i Villodre et Richard Wicentowski, éditeurs : *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval@ACL 2007, Prague, Czech Republic, June 23-24, 2007*,

pages 398–401. The Association for Computer Linguistics, 2007. URL http://aclweb.org/anthology/S/S07/S07-1088.pdf.

Jöran Beel, Bela Gipp, Stefan Langer et Corinna Breitinger : Research-paper recommender systems : a literature survey. *Int. J. on Digital Libraries*, 17(4):305–338, 2016. URL https://doi.org/10.1007/s00799-015-0156-0.

Alejandro Bellogín, Iván Cantador et Pablo Castells : A comparative study of heterogeneous item recommendations in social systems. *Inf. Sci.*, 221:142–169, 2013. URL https://doi.org/10.1016/j.ins.2012.09.039.

Alejandro Bellogín, Pablo Castells et Iván Cantador : Precision-oriented evaluation of recommender systems : an algorithmic comparison. *In* Bamshad Mobasher, Robin D. Burke, Dietmar Jannach et Gediminas Adomavicius, éditeurs : *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pages 333–336. ACM, 2011. URL http://doi.acm.org/10.1145/2043932.2043996.

James Bennett et Stan Lanning : The netflix prize. *In In KDD Cup and Workshop in conjunction with KDD*, 2007.

Tim Berners-Lee, James Hendler et Ora Lassila : The semantic web. *Scientific American Magazine*, pages 29–37, 5 2001.

Daniel Billsus et Michael J. Pazzani : Learning collaborative information filters. *In* Jude W. Shavlik, éditeur : *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998*, pages 46–54. Morgan Kaufmann, 1998.

Daniel Billsus et Michael J. Pazzani : A personal news agent that talks, learns and explains. *In* Oren Etzioni, Jörg P. Müller et Jeffrey M. Bradshaw, éditeurs : *Proceedings of the Third Annual Conference on Autonomous Agents, AGENTS 1999, Seattle, WA, USA, May 1-5, 1999*, pages 268–275. ACM, 1999. URL http://doi.acm.org/10.1145/301136.301208.

Daniel Billsus et Michael J. Pazzani : User modeling for adaptive news access. *User Model. User-Adapt. Interact.*, 10(2-3):147–180, 2000. URL https://doi.org/10.1023/A:1026501525781.

David M. Blei, Andrew Y. Ng et Michael I. Jordan : Latent dirichlet allocation. *In* Thomas G. Dietterich, Suzanna Becker et Zoubin Ghahramani, éditeurs : *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems : Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 601–608. MIT Press, 2001. URL http://papers.nips.cc/paper/2070-latent-dirichlet-allocation.

Rubi Boim, Tova Milo et Slava Novgorodov : Diversification and refinement in collaborative filtering recommender. *In* Craig Macdonald, Iadh Ounis et Ian Ruthven, éditeurs : *Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, 2011*, pages 739–744. ACM, 2011. URL http://doi.acm.org/10.1145/2063576.2063684.

Allan Borodin, Hyun Chul Lee et Yuli Ye : Max-sum diversification, monotone submodular functions and dynamic updates. pages 155–166, 2012. URL http://doi.acm.org/10.1145/2213556.2213580.

Gerlof Bouma : Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40, 2009.

Pierre Bourdieu : *Le sens pratique*. Le Sens commun. Editions de Minuit, Paris, France, 1980. URL https://books.google.fr/books?id=iUvuZwEACAAJ.

Keith Bradley et Barry Smyth : Improving recommendation diversity. *In Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science, Maynooth, Ireland*, pages 85–94, 2001.

Karlheinz Brandenburg, Christian Dittmar, Matthias Gruhne, Jakob Abesser, Hanna Lukashevich, Peter Dunker, Daniel Gärtner, Kay Wolter et Holger Grossmann : Music search and recommendation. *In* Borko Furht, éditeur : *Handbook of Multimedia for Digital Entertainment and Arts*, pages 349–384. Springer US, Boston, MA, 2009. URL https://doi.org/10.1007/978-0-387-89024-1_16.

John S. Breese, David Heckerman et Carl Myers Kadie : Empirical analysis of predictive algorithms for collaborative filtering. *In* Gregory F. Cooper et Serafín Moral, éditeurs : *UAI '98 : Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, July 24-26, 1998*, pages 43–52. Morgan Kaufmann, 1998. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=231&proceeding_id=14.

Robin D. Burke : Hybrid recommender systems : Survey and experiments. *User Model. User-Adapt. Interact.*, 12(4):331–370, 2002. URL https://doi.org/10.1023/A:1021240730564.

Robin D. Burke : Hybrid web recommender systems. *In* Peter Brusilovsky, Alfred Kobsa et Wolfgang Nejdl, éditeurs : *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 de *Lecture Notes in Computer Science*, pages 377–408. Springer, 2007. URL https://doi.org/10.1007/978-3-540-72079-9_12.

Tadeusz Calinski et J. Harabasz : A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

Iván Cantador, Alejandro Bellogín et David Vallet : Content-based recommendation in social tagging systems. *In* Xavier Amatriain, Marc Torrens, Paul Resnick et Markus Zanker, éditeurs : *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 237–240. ACM, 2010. URL http://doi.acm.org/10.1145/1864708.1864756.

Iván Cantador, Peter Brusilovsky et Tsvi Kuflik : 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). *In Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

Jaime G. Carbonell et Jade Goldstein : The use of mmr, diversity-based reranking for reordering documents and producing summaries. *In* W. Bruce Croft, Alistair Moffat, C. J. van Rijsbergen, Ross Wilkinson et Justin Zobel, éditeurs : *SIGIR '98 : Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, pages 335–336. ACM, 1998. URL http://doi.acm.org/10.1145/290941.291025.

Dominique Cardon : *A quoi rêvent les algorithmes : Nos vies à l'heure des big data*. Coédition Seuil-La République des idées. Seuil, Paris, France, 2015.

Sylvain Castagnos, Armelle Brun et Anne Boyer : Utilité et perception de la diversité dans les systèmes de recommandation. *In* Catherine Berrut, éditeur : *CORIA 2013 - Conférence en Recherche d'Infomations et Applications - 10th French Information Retrieval Conference, Neuchâtel, Suisse, April 3-5, 2013.*, pages 237–252. UNINE, 2013. URL https://doi.org/10.24348/coria.2013.coria2013_47.

Pablo Castells, Neil J. Hurley et Saul Vargas : Novelty and diversity in recommender systems. *In* Francesco Ricci, Lior Rokach et Bracha Shapira, éditeurs : *Recommender Systems Handbook*, pages 881–918. Springer, 2015. URL https://doi.org/10.1007/978-1-4899-7637-6_26.

Òscar Celma et Pedro Cano : From hits to niches ? : Or how popular artists can bias music recommendation and discovery. *In Proceedings of the 2Nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, NETFLIX '08, pages 5 :1–5 :8, New York, NY, USA, 2008. ACM. URL http://doi.acm.org/10.1145/1722149.1722154.

Òscar Celma et Perfecto Herrera : A new approach to evaluating novel recommendations. *In* Pearl Pu, Derek G. Bridge, Bamshad Mobasher et Francesco Ricci, éditeurs : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 179–186. ACM, 2008. URL http://doi.acm.org/10.1145/1454008.1454038.

Timothy M. Chan : Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, 16(4):361–368, 1996. URL https://doi.org/10.1007/BF02712873.

Sushma Channamsetty et Michael D. Ekstrand : Recommender response to diversity and popularity bias in user profiles. *In* Vasile Rus et Zdravko Markov, éditeurs : *Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2017, Marco Island, Florida, USA, May 22-24, 2017.*, pages 657–660. AAAI Press, 2017. URL https://aaai.org/ocs/index.php/FLAIRS/FLAIRS17/paper/view/15524.

Olivier Chapelle, Shihao Ji, Ciya Liao, Emre Velipasaoglu, Larry Lai et Su-Lin Wu : Intent-based diversification of web search results : metrics and algorithms. *Inf. Retr.*, 14(6):572–592, 2011. URL https://doi.org/10.1007/s10791-011-9167-7.

Harr Chen et David R. Karger : Less is more : probabilistic models for retrieving fewer relevant documents. *In* Efthimis N. Efthimiadis, Susan T. Dumais, David Hawking et Kalervo Järvelin, éditeurs : *SIGIR 2006 : Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*, pages 429–436. ACM, 2006. URL http://doi.acm.org/10.1145/1148170.1148245.

Liren Chen et Katia P. Sycara : Webmate : A personal agent for browsing and searching. *In* Katia P. Sycara et Michael Wooldridge, éditeurs : *Proceedings of the Second International Conference on Autonomous Agents, AGENTS 1998, St. Paul, Minnepolis, USA, May 9-13, 1998*, pages 132–139. ACM, 1998. URL http://doi.acm.org/10.1145/280765.280789.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu et Hemal Shah : Wide & deep learning for recommender systems. *CoRR*, abs/1606.07792, 2016. URL http://arxiv.org/abs/1606.07792.

Keunwoo Choi, György Fazekas et Mark B. Sandler : Automatic tagging using deep convolutional neural networks. *In* Michael I. Mandel, Johanna Devaney, Douglas Turnbull et George Tzanetakis, éditeurs : *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, pages 805–811, 2016. URL https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/009_Paper.pdf.

Cédric Claquin et Pierre-René Lhérisson : Au service de la création indépendante. *In* Philippe Le Guern, éditeur : *Où va la musique ? Numérimorphose et nouvelles expériences d'écoute*, pages 153–166. Presse des Mines, Libres opinions, Paris, France, 2016. URL https://www.pressesdesmines.com/produit/ou-va-la-musique/.

Paul Covington, Jay Adams et Emre Sargin : Deep neural networks for youtube recommendations. *In* Shilad Sen, Werner Geyer, Jill Freyne et Pablo Castells, éditeurs : *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 191–198. ACM, 2016. URL http://doi.acm.org/10.1145/2959100.2959190.

Paolo Cremonesi, Yehuda Koren et Roberto Turrin : Performance of recommender algorithms on top-n recommendation tasks. *In* Xavier Amatriain, Marc Torrens, Paul Resnick et Markus Zanker, éditeurs : *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 39–46. ACM, 2010. URL http://doi.acm.org/10.1145/1864708.1864721.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas et Richard A. Harshman : Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990. URL https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9.

Mukund Deshpande et George Karypis : Item-based top-*N* recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004. URL http://doi.acm.org/10.1145/963770.963776.

Dennis Diefenbach, Pierre-René Lhérisson, Fabrice Muhlenbach et Pierre Maret : Computing the semantic relatedness of music genre using semantic web data. *In* Michael Martin, Martí Cuquet et Erwin Folmer, éditeurs : *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016.*, volume 1695 de *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. URL http://ceur-ws.org/Vol-1695/paper23.pdf.

Michael D. Ekstrand, F. Maxwell Harper, Martijn C. Willemsen et Joseph A. Konstan : User perception of differences in recommender algorithms. *In* Alfred Kobsa, Michelle X. Zhou, Martin Ester et Yehuda Koren, éditeurs : *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, pages 161–168. ACM, 2014. URL http://doi.acm.org/10.1145/2645710.2645737.

Bruce Ferwerda, Mark P. Graus, Andreu Vall, Marko Tkalcic et Markus Schedl : How item discovery enabled by diversity leads to increased recommendation list attractiveness. *In* Ahmed Seffah, Birgit Penzenstadler, Carina Alves et Xin Peng, éditeurs : *Proceedings of the Symposium on Applied Computing, SAC 2017, Marrakech, Morocco, April 3-7, 2017*, pages 1693–1696. ACM, 2017. URL http://doi.acm.org/10.1145/3019612.3019899.

Daniel Fleder et Kartik Hosanagar : Blockbuster culture's next rise or fall : The impact of recommender systems on sales diversity. *Management science*, 55(5):697–712, 2009a.

Daniel M. Fleder et Kartik Hosanagar : Blockbuster culture's next rise or fall : The impact of recommender systems on sales diversity. *Management Science*, 55(5):697–712, 2009b. URL https://doi.org/10.1287/mnsc.1080.0974.

Tim Furche, Georg Gottlob, Giovanni Grasso, Xiaonan Guo, Giorgio Orsi, Christian Schallhart et Cheng Wang : DIADEM : thousands of websites to a single database. *PVLDB*, 7 (14):1845–1856, 2014.

Mouzhi Ge, Carla Delgado-Battenfeld et Dietmar Jannach : Beyond accuracy : evaluating recommender systems by coverage and serendipity. *In* Xavier Amatriain, Marc Torrens, Paul Resnick et Markus Zanker, éditeurs : *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 257–260. ACM, 2010. URL http://doi.acm.org/10.1145/1864708.1864761.

Mouzhi Ge, Dietmar Jannach, Fatih Gedikli et Martin Hepp : Effects of the placement of diverse items in recommendation lists. *In* Leszek A. Maciaszek, Alfredo Cuzzocrea et José Cordeiro, éditeurs : *ICEIS 2012 - Proceedings of the 14th International Conference on Enterprise Information Systems, Volume 2, Wroclaw, Poland, 28 June - 1 July, 2012*, pages 201–208. SciTePress, 2012.

Marco De Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci et Giovanni Semeraro : Semantics-aware content-based recommender systems. *In* Francesco Ricci, Lior Rokach et Bracha Shapira, éditeurs : *Recommender Systems Handbook*, pages 119–159. Springer, 2015a. URL https://doi.org/10.1007/978-1-4899-7637-6_4.

Marco De Gemmis, Pasquale Lops, Giovanni Semeraro et Pierpaolo Basile : Integrating tags in a semantic content-based recommender. *In* Pearl Pu, Derek G. Bridge, Bamshad Mobasher et Francesco Ricci, éditeurs : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 163–170. ACM, 2008. URL http://doi.acm.org/10.1145/1454008.1454036.

Marco De Gemmis, Pasquale Lops, Giovanni Semeraro et Cataldo Musto : An investigation on the serendipity problem in recommender systems. *Inf. Process. Manage.*, 51(5):695–717, 2015b. URL https://doi.org/10.1016/j.ipm.2015.06.008.

David Goldberg, David A. Nichols, Brian M. Oki et Douglas B. Terry : Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992. URL http://doi.acm.org/10.1145/138859.138867.

Karthik Gomadam, Peter Z. Yeh et Kunal Verma : Data enrichment using data sources on the web. *In Intelligent Web Services Meet Social Computing, Papers from the 2012 AAAI Spring Symposium, Palo Alto, California, USA, March 26-28, 2012*, volume SS-12-04 de

*AAAI Technical Report*. AAAI, 2012. URL http://www.aaai.org/ocs/index.php/SSS/SSS12/paper/view/4336.

Ronald L. Graham : An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Process. Lett.*, 1(4):132–133, 1972. URL https://doi.org/10.1016/0020-0190(72)90045-2.

Felix Grässer, Stefanie Beckert, Denise Küster, Jochen Schmitt, Susanne Abraham, Hagen Malberg et Sebastian Zaunseder : Therapy decision support based on recommender system methods. *Journal of Healthcare Engineering*, 2017, 2017.

Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan et Doug Sharp : E-commerce in your inbox : Product recommendations at scale. *In* Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu et Graham Williams, éditeurs : *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1809–1818. ACM, 2015. URL http://doi.acm.org/10.1145/2783258.2788627.

Stephen J. Green, Paul Lamere, Jeffrey Alexander, François Maillet, Susanna Kirk, Jessica Holt, Jackie Bourque et Xiao-Wen Mak : Generating transparent, steerable recommendations from textual descriptions of items. *In* Lawrence D. Bergman, Alexander Tuzhilin, Robin D. Burke, Alexander Felfernig et Lars Schmidt-Thieme, éditeurs : *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009*, pages 281–284. ACM, 2009. URL http://doi.acm.org/10.1145/1639714.1639768.

Thomas R. Gruber : A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, juin 1993. ISSN 1042-8143. URL http://dx.doi.org/10.1006/knac.1993.1008.

Asela Gunawardana et Guy Shani : A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, 10:2935–2962, 2009. URL http://doi.acm.org/10.1145/1577069.1755883.

Asela Gunawardana et Guy Shani : Evaluating recommender systems. *In* Francesco Ricci, Lior Rokach et Bracha Shapira, éditeurs : *Recommender Systems Handbook*, pages 265–308. Springer, 2015. URL https://doi.org/10.1007/978-1-4899-7637-6_8.

Shubhanshu Gupta : Music data analysis : A state-of-the-art survey. *CoRR*, abs/1411.5014, 2014. URL http://arxiv.org/abs/1411.5014.

F. Maxwell Harper et Joseph A. Konstan : The movielens datasets : History and context. *TiiS*, 5 (4):19 :1–19 :19, 2016. URL http://doi.acm.org/10.1145/2827872.

Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers et John Riedl : An algorithmic framework for performing collaborative filtering. *In* Fredric C. Gey, Marti A. Hearst et Richard M. Tong, éditeurs : *SIGIR '99 : Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999, Berkeley, CA, USA*, pages 230–237. ACM, 1999. URL http://doi.acm.org/10.1145/312624.312682.

Jonathan L. Herlocker, Joseph A. Konstan et John Riedl : An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr.*, 5(4):287–310, 2002. URL https://doi.org/10.1023/A:1020443909834.

Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen et John Riedl : Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004. URL http://doi.acm.org/10.1145/963770.963772.

Matthew Hindman : *The myth of digital democracy*. Princeton University Press, Princeton, New Jersey, USA, 2008.

Jinwon Ho et Rong Tang : Towards an optimal resolution to information overload : an infomediary approach. *In Proceedings of GROUP 2001, ACM 2001 International Conference on Supporting Group Work, September 30 - October 3, 2001, Boulder, Colorado, USA*, pages 91–96. ACM, 2001. URL http://doi.acm.org/10.1145/500286.500302.

Kartik Hosanagar, Daniel M. Fleder, Dokyun Lee et Andreas Buja : Will the global village fracture into tribes ? recommender systems and their effects on consumer fragmentation. *Management Science*, 60(4):805–823, 2014. URL https://doi.org/10.1287/mnsc.2013.1808.

Rong Hu et Pearl Pu : Enhancing recommendation diversity with organization interfaces. *In* Pearl Pu, Michael J. Pazzani, Elisabeth André et Doug Riecken, éditeurs : *Proceedings of the 16th International Conference on Intelligent User Interfaces, IUI 2011, Palo Alto, CA, USA, February 13-16, 2011*, pages 347–350. ACM, 2011. URL http://doi.acm.org/10.1145/1943403.1943462.

Yifan Hu, Yehuda Koren et Chris Volinsky : Collaborative filtering for implicit feedback datasets. *In Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pages 263–272. IEEE Computer Society, 2008. URL https://doi.org/10.1109/ICDM.2008.22.

Anna Huang : Similarity measures for text document clustering. *In Proceedings of the 6th New Zealand Computer Science Research Student Conference (NZCSRSC 2008), April 14-18, 2008, Christchurch, New Zealand*, pages 49–56, 2008.

Leo Iaquinta, Marco De Gemmis, Pasquale Lops, Giovanni Semeraro, Michele Filannino et Piero Molino : Introducing serendipity in a content-based recommender system. *In* Fatos Xhafa, Francisco Herrera, Ajith Abraham, Mario Köppen et José Manuel Benítez, éditeurs : *8th International Conference on Hybrid Intelligent Systems (HIS 2008), September 10-12, 2008, Barcelona, Spain*, pages 168–173. IEEE Computer Society, 2008. URL https://doi.org/10.1109/HIS.2008.25.

Prateek Jain, Pascal Hitzler, Amit P. Sheth, Kunal Verma et Peter Z. Yeh : Ontology alignment for linked open data. *In* Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks et Birte Glimm, éditeurs : *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I*, volume 6496 de *Lecture Notes in Computer Science*, pages 402–417. Springer, 2010. URL https://doi.org/10.1007/978-3-642-17746-0_26.

Dietmar Jannach et Gediminas Adomavicius : Recommendations with a purpose. *In* Shilad Sen, Werner Geyer, Jill Freyne et Pablo Castells, éditeurs : *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 7–10. ACM, 2016. URL http://doi.acm.org/10.1145/2959100.2959186.

Dietmar Jannach, Lukas Lerche, Fatih Gedikli et Geoffray Bonnin : What recommenders recommend - an analysis of accuracy, popularity, and sales diversity effects. *In* Sandra Carberry, Stephan Weibelzahl, Alessandro Micarelli et Giovanni Semeraro, éditeurs : *User Modeling, Adaptation, and Personalization - 21th International Conference, UMAP 2013, Rome, Italy, June 10-14, 2013, Proceedings*, volume 7899 de *Lecture Notes in Computer Science*, pages 25–37. Springer, 2013. URL https://doi.org/10.1007/978-3-642-38844-6_3.

Kalervo Järvelin et Jaana Kekäläinen : Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002. URL http://doi.acm.org/10.1145/582415.582418.

Ray A. Jarvis : On the identification of the convex hull of a finite set of points in the plane. *Inf. Process. Lett.*, 2(1):18–21, 1973. URL https://doi.org/10.1016/0020-0190(73)90020-3.

Ajita John et Dorée Seligmann : Collaborative tagging and expertise in the enterprise. *In Proc. of the Collaborative Web Tagging Workshop at WWW 2006, May 22-26, 2006, Edinburgh, UK*, 2006.

Marius Kaminskas et Derek Bridge : Diversity, serendipity, novelty, and coverage : A survey and empirical analysis of beyond-accuracy objectives in recommender systems. *TiiS*, 7 (1):2 :1–2 :42, 2017. URL http://doi.acm.org/10.1145/2926720.

Alexandros Karatzoglou, Linas Baltrunas et Yue Shi : Learning to rank for recommender systems. *In* Qiang Yang, Irwin King, Qing Li, Pearl Pu et George Karypis, éditeurs : *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 493–494. ACM, 2013. URL http://doi.acm.org/10.1145/2507157.2508063.

Mohammad Khabbaz et Laks V. S. Lakshmanan : Toprecs : Top-k algorithms for item-based collaborative filtering. *In* Anastasia Ailamaki, Sihem Amer-Yahia, Jignesh M. Patel, Tore Risch, Pierre Senellart et Julia Stoyanovich, éditeurs : *EDBT 2011, 14th International Conference on Extending Database Technology, Uppsala, Sweden, March 21-24, 2011, Proceedings*, pages 213–224. ACM, 2011. URL http://doi.acm.org/10.1145/1951365.1951392.

Dohyun Kim et Bong-Jin Yum : Collaborative filtering based on iterative principal component analysis. *Expert Syst. Appl.*, 28(4):823–830, 2005. URL https://doi.org/10.1016/j.eswa.2004.12.037.

Teuvo Kohonen : *Self-Organizing Maps*. Springer Series in Information Sciences. Springer, Berlin Heidelberg, Germany, third édition, 2001.

Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon et John Riedl : Grouplens : Applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, 1997. URL http://doi.acm.org/10.1145/245108.245126.

Yehuda Koren : Factorization meets the neighborhood : a multifaceted collaborative filtering model. *In* Ying Li, Bing Liu et Sunita Sarawagi, éditeurs : *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pages 426–434. ACM, 2008. URL http://doi.acm.org/10.1145/1401890.1401944.

Yehuda Koren : Collaborative filtering with temporal dynamics. *Commun. ACM*, 53(4):89–97, 2010. URL http://doi.acm.org/10.1145/1721654.1721677.

Yehuda Koren, Robert M. Bell et Chris Volinsky : Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009. URL https://doi.org/10.1109/MC.2009.263.

Maciej Kula : Metadata embeddings for user and item cold-start recommendations. *In* Toine Bogers et Marijn Koolen, éditeurs : *Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015.*, volume 1448 de *CEUR Workshop Proceedings*, pages 14–21. CEUR-WS.org, 2015. URL http://ceur-ws.org/Vol-1448/paper4.pdf.

Quoc V. Le et Tomas Mikolov : Distributed representations of sentences and documents. *In Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 de *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014. URL http://jmlr.org/proceedings/papers/v32/le14.html.

Yann LeCun, Yoshua Bengio et Geoffrey E. Hinton : Deep learning. *Nature*, 521(7553):436–444, 2015.

Kibeom Lee et Kyogu Lee : Using experts among users for novel movie recommendations. *JCSE*, 7(1):21–29, 2013. URL https://doi.org/10.5626/JCSE.2013.7.1.21.

Pierre-René Lhérisson, Fabrice Muhlenbach et Pierre Maret : Application mobile pour l'évaluation d'un algorithme de calcul de distance entre des items musicaux. *In* Fabien L. Gandon et Gilles Bisson, éditeurs : *17ème Journées Francophones Extraction et Gestion des Connaissances, EGC 2017, 24-27 Janvier 2017, Grenoble, France*, volume E-33 de *RNTI*, pages 461–464. Éditions RNTI, 2017a. URL http://editions-rnti.fr/?inprocid=1002324.

Pierre-René Lhérisson, Fabrice Muhlenbach et Pierre Maret : Fair recommendations through diversity promotion. *In* Gao Cong, Wen-Chih Peng, Wei Emma Zhang, Chengliang Li et Aixin Sun, éditeurs : *Advanced Data Mining and Applications - 13th International Conference, ADMA 2017, Singapore, November 5-6, 2017, Proceedings*, volume 10604 de *Lecture Notes in Computer Science*, pages 89–103. Springer, 2017b. ISBN 978-3-319-69178-7. URL https://doi.org/10.1007/978-3-319-69179-4_7.

Pierre-René Lhérisson, Fabrice Muhlenbach et Pierre Maret : Recommandations et prédictions de préférences basées sur la combinaison de données sémantiques et de folksonomie. *In* Fabien L. Gandon et Gilles Bisson, éditeurs : *17ème Journées Francophones Extraction et Gestion des Connaissances, EGC 2017, 24-27 Janvier 2017, Grenoble, France*, volume E-33 de *RNTI*, pages 333–338. Éditions RNTI, 2017c. URL http://editions-rnti.fr/?inprocid=1002294.

Xiaohui Li et Tomohiro Murata : Using multidimensional clustering based collaborative filtering approach improving recommendation diversity. *In 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Macau, China, December 4-7, 2012*, pages 169–174. IEEE Computer Society, 2012. URL https://doi.org/10.1109/WI-IAT.2012.229.

Henry Lieberman : Letizia : An agent that assists web browsing. *In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 924–929. Morgan Kaufmann, 1995. URL http://ijcai.org/Proceedings/95-1/Papers/119.pdf.

Daryl Lim, Gert R. G. Lanckriet et Brian McFee : Robust structural metric learning. *In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 de *JMLR Workshop and Conference Proceedings*, pages 615–623. JMLR.org, 2013. URL http://jmlr.org/proceedings/papers/v28/lim13.html.

Greg Linden, Brent Smith et Jeremy York : Amazon.com recommendations : Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003. URL https://doi.org/10.1109/MIC.2003.1167344.

Nathan Nan Liu, Min Zhao, Evan Wei Xiang et Qiang Yang : Online evolutionary collaborative filtering. *In* Xavier Amatriain, Marc Torrens, Paul Resnick et Markus Zanker, éditeurs : *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pages 95–102. ACM, 2010. URL http://doi.acm.org/10.1145/1864708.1864729.

Tie-Yan Liu : Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009. URL https://doi.org/10.1561/1500000016.

Xin Liu et Karl Aberer : Towards a dynamic top-n recommendation framework. *In* Alfred Kobsa, Michelle X. Zhou, Martin Ester et Yehuda Koren, éditeurs : *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, pages 217–224. ACM, 2014. URL http://doi.acm.org/10.1145/2645710.2645720.

Pasquale Lops, Marco De Gemmis et Giovanni Semeraro : Content-based recommender systems : State of the art and trends. *In* Francesco Ricci, Lior Rokach, Bracha Shapira et Paul B. Kantor, éditeurs : *Recommender Systems Handbook*, pages 73–105. Springer, 2011. URL https://doi.org/10.1007/978-0-387-85820-3_3.

Michael Lubatkin et Sayan Chatterjee : Extending modern portfolio theory into the domain of corporate diversification : does it apply ? *Academy of Management Journal*, 37(1):109–136, 1994.

Hao Ma, Irwin King et Michael R. Lyu : Effective missing data prediction for collaborative filtering. *In* Wessel Kraaij, Arjen P. de Vries, Charles L. A. Clarke, Norbert Fuhr et Noriko Kando, éditeurs : *SIGIR 2007 : Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 39–46. ACM, 2007. URL http://doi.acm.org/10.1145/1277741.1277751.

Valentina Maccatrozzo, Manon Terstall, Lora Aroyo et Guus Schreiber : SIRUP : serendipity in recommendations via user perceptions. *In* George A. Papadopoulos, Tsvi Kuflik, Fang Chen,

Carlos Duarte et Wai-Tat Fu, éditeurs : *Proceedings of the 22nd International Conference on Intelligent User Interfaces, IUI 2017, Limassol, Cyprus, March 13-16, 2017*, pages 35–44. ACM, 2017. URL http://doi.acm.org/10.1145/3025171.3025185.

Terence Magno et Carl Sable : A comparison of signal based music recommendation to genre labels, collaborative filtering, musicological analysis, human recommendation and random baseline. *In* Juan Pablo Bello, Elaine Chew et Douglas Turnbull, éditeurs : *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, pages 161–166, 2008. URL http://ismir2008.ismir.net/papers/ISMIR2008_157.pdf.

David Marr et Ellen C. Hildreth : Theory of edge detection. *Proceedings of The Royal Society. Series B, Biological Sciences*, 207(1167):187–217, February 1980.

Leigh McAlister et Edgar Pessemier : Variety seeking behavior : An interdisciplinary review. *Journal of Consumer Research*, 9(3):311–322, 1982.

Julian J. McAuley, Christopher Targett, Qinfeng Shi et Anton van den Hengel : Image-based recommendations on styles and substitutes. *In* Ricardo A. Baeza-Yates, Mounia Lalmas, Alistair Moffat et Berthier A. Ribeiro-Neto, éditeurs : *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, pages 43–52. ACM, 2015. URL http://doi.acm.org/10.1145/2766462.2767755.

Brian McFee, Thierry Bertin-Mahieux, Daniel P. W. Ellis et Gert R. G. Lanckriet : The million song dataset challenge. *In* Alain Mille, Fabien L. Gandon, Jacques Misselis, Michael Rabinovich et Steffen Staab, éditeurs : *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*, pages 909–916. ACM, 2012. URL http://doi.acm.org/10.1145/2187980.2188222.

Sean M. McNee, John Riedl et Joseph A. Konstan : Being accurate is not enough : how accuracy metrics have hurt recommender systems. *In* Gary M. Olson et Robin Jeffries, éditeurs : *Extended Abstracts Proceedings of the 2006 Conference on Human Factors in Computing Systems, CHI 2006, Montréal, Québec, Canada, April 22-27, 2006*, pages 1097–1101. ACM, 2006. URL http://doi.acm.org/10.1145/1125451.1125659.

Kevin Mellet : Aux sources du marketing viral. *Réseaux*, 157–158(5):267–292, 2009.

Alan Menk, Laura Sebastia et Rebeca Ferreira : Curumim : A serendipitous recommender system based on human curiosity. *In* Cecilia Zanni-Merk, Claudia S. Frydman, Carlos Toro, Yulia Hicks, Robert J. Howlett et Lakhmi C. Jain, éditeurs : *Knowledge-Based and Intelligent Information & Engineering Systems : Proceedings of the 21st International Conference KES-2017, Marseille, France, 6-8 September 2017.*, volume 112 de *Procedia Computer Science*,

pages 484–493. Elsevier, 2017. URL https://doi.org/10.1016/j.procs.2017.08.098.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado et Jeffrey Dean : Distributed representations of words and phrases and their compositionality. *In* Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani et Kilian Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013. URL http://tinyurl.com/y9y4lesm.

Alexandros Moukas : Amalthaea information discovery and filtering using a multiagent evolving ecosystem. *Applied Artificial Intelligence*, 11(5):437–457, 1997. URL https://doi.org/10.1080/088395197118127.

Fabrice Muhlenbach, Pierre-René Lhérisson et Pierre Maret : Procédé de sélection automatique d'un contenu multimédia dans une base de données. Brevet d'invention FR3046269, INPI, 2017.

Tomoko Murakami, Koichiro Mori et Ryohei Orihara : Metrics for evaluating the serendipity of recommendation lists. *In* Ken Satoh, Akihiro Inokuchi, Katashi Nagao et Takahiro Kawamura, éditeurs : *New Frontiers in Artificial Intelligence, JSAI 2007 Conference and Workshops, Miyazaki, Japan, June 18-22, 2007, Revised Selected Papers*, volume 4914 de *Lecture Notes in Computer Science*, pages 40–46. Springer, 2007. URL https://doi.org/10.1007/978-3-540-78197-4_5.

Mark A. Musen : Dimensions of knowledge sharing and reuse. *Computers and Biomedical Research*, 25(5):435 – 467, 1992. ISSN 0010-4809. URL http://www.sciencedirect.com/science/article/pii/001048099290003S.

Cataldo Musto, Pierpaolo Basile, Pasquale Lops, Marco de Gemmis et Giovanni Semeraro : Introducing linked open data in graph-based recommender systems. *Inf. Process. Manage.*, 53 (2):405–435, 2017. URL https://doi.org/10.1016/j.ipm.2016.12.003.

Cataldo Musto, Claudio Greco, Alessandro Suglia et Giovanni Semeraro : Ask me any rating : A content-based recommender system based on recurrent neural networks. *In* Giorgio Maria Di Nunzio, Franco Maria Nardini et Salvatore Orlando, éditeurs : *Proceedings of the 7th Italian Information Retrieval Workshop, Venezia, Italy, May 30-31, 2016.*, volume 1653 de *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. URL http://ceur-ws.org/Vol-1653/paper_11.pdf.

Cataldo Musto, Giovanni Semeraro, Marco De Gemmis et Pasquale Lops : Word embedding techniques for content-based recommender systems : An empirical evaluation. *In* Pablo Castells, éditeur : *Poster Proceedings of the 9th ACM Conference on Recommender Systems, RecSys*

*2015, Vienna, Austria, September 16, 2015.*, volume 1441 de *CEUR Workshop Proceedings*. CEUR-WS.org, 2015. URL http://ceur-ws.org/Vol-1441/recsys2015_poster23.pdf.

Makoto Nakatsuji, Yasuhiro Fujiwara, Akimichi Tanaka, Toshio Uchiyama, Ko Fujimura et Toru Ishida : Classical music for rock fans ? : Novel recommendations for expanding user interests. *In* Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson et Aijun An, éditeurs : *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 949–958. ACM, 2010. URL http://doi.acm.org/10.1145/1871437.1871558.

Nagarajan Natarajan, Donghyuk Shin et Inderjit S. Dhillon : Which app will you use next ? : collaborative filtering with interactional context. *In* Qiang Yang, Irwin King, Qing Li, Pearl Pu et George Karypis, éditeurs : *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 201–208. ACM, 2013. URL http://doi.acm.org/10.1145/2507157.2507186.

Tien T. Nguyen, Pik-Mai Hui, F. Maxwell Harper, Loren G. Terveen et Joseph A. Konstan : Exploring the filter bubble : the effect of using recommender systems on content diversity. *In* Chin-Wan Chung, Andrei Z. Broder, Kyuseok Shim et Torsten Suel, éditeurs : *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, pages 677–686. ACM, 2014. URL http://doi.acm.org/10.1145/2566486.2568012.

Tommaso Di Noia, Roberto Mirizzi, Vito Claudio Ostuni, Davide Romito et Markus Zanker : Linked open data to support content-based recommender systems. *In* Valentina Presutti et Helena Sofia Pinto, éditeurs : *I-SEMANTICS 2012 - 8th International Conference on Semantic Systems, I-SEMANTICS '12, Graz, Austria, September 5-7, 2012*, pages 1–8. ACM, 2012. URL http://doi.acm.org/10.1145/2362499.2362501.

Tommaso Di Noia, Jessica Rosati, Paolo Tomeo et Eugenio Di Sciascio : Adaptive multi-attribute diversity for recommender systems. *Inf. Sci.*, 382-383:234–253, 2017. URL https://doi.org/10.1016/j.ins.2016.11.015.

Sergio Oramas, Oriol Nieto, Mohamed Sordo et Xavier Serra : A deep multimodal approach for cold-start music recommendation. *In* Balázs Hidasi, Alexandros Karatzoglou, Oren Sar Shalom, Sander Dieleman, Bracha Shapira et Domonkos Tikk, éditeurs : *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2017, Como, Italy, August 27, 2017*, pages 32–37. ACM, 2017. URL http://doi.acm.org/10.1145/3125486.3125492.

Sergio Oramas, Mohamed Sordo, Luis Espinosa Anke et Xavier Serra : A semantic-based approach for artist similarity. *In* Meinard Müller et Frans Wiering, éditeurs : *Proceedings*

*of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, pages 100–106, 2015. URL http://ismir2015.uma.es/articles/305_Paper.pdf.

Humberto Jesús Corona Pampín, Houssem Jerbi et Michael P. O'Mahony : Evaluating the relative performance of collaborative filtering recommender systems. *J. UCS*, 21(13):1849–1868, 2015. URL http://www.jucs.org/jucs_21_13/evaluating_the_relative_performance.

Shameem Puthiya Parambath, Nicolas Usunier et Yves Grandvalet : A coverage-based approach to recommendation diversity on similarity graph. *In* Shilad Sen, Werner Geyer, Jill Freyne et Pablo Castells, éditeurs : *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 15–22. ACM, 2016. URL http://doi.acm.org/10.1145/2959100.2959149.

Elie Pariser : *The Filter Bubble : What The Internet Is Hiding From You*. Penguin Press, New York, NY, USA, 2011.

Hyunjung Park et Jennifer Widom : Crowdfill : collecting structured data from the crowd. *In International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 577–588, 2014.

Yoon-Joo Park et Alexander Tuzhilin : The long tail of recommender systems and how to leverage it. *In* Pearl Pu, Derek G. Bridge, Bamshad Mobasher et Francesco Ricci, éditeurs : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 11–18. ACM, 2008. URL http://doi.acm.org/10.1145/1454008.1454012.

Denis Parra et Shaghayegh Sahebi : Recommender systems : Sources of knowledge and evaluation metrics. *In* Juan D. Velásquez, Vasile Palade et Lakhmi C. Jain, éditeurs : *Advanced Techniques in Web Intelligence-2 : Web User Browsing Behaviour and Preference Analysis*, pages 149–175. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. URL https://doi.org/10.1007/978-3-642-33326-2_7.

Alexandre Passant : dbrec - music recommendations using dbpedia. *In* Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks et Birte Glimm, éditeurs : *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part II*, volume 6497 de *Lecture Notes in Computer Science*, pages 209–224. Springer, 2010. URL https://doi.org/10.1007/978-3-642-17749-1_14.

G. P. Patil et C. Taillie : Diversity as a concept and its measurement. *Journal of the American*

*Statistical Association*, 77(379):548–561, 1982. ISSN 01621459. URL http://www.jstor.org/stable/2287709.

Michael J. Pazzani et Daniel Billsus : Content-based recommendation systems. *In* Peter Brusilovsky, Alfred Kobsa et Wolfgang Nejdl, éditeurs : *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 de *Lecture Notes in Computer Science*, pages 325–341. Springer, 2007. URL https://doi.org/10.1007/978-3-540-72079-9_10.

Michael J. Pazzani, Jack Muramatsu et Daniel Billsus : Syskill & webert : Identifying interesting web sites. *In* William J. Clancey et Daniel S. Weld, éditeurs : *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, August 4-8, 1996, Volume 1.*, pages 54–61. AAAI Press / The MIT Press, 1996. URL http://www.aaai.org/Library/AAAI/1996/aaai96-008.php.

Jeffrey Pennington, Richard Socher et Christopher D. Manning : Glove : Global vectors for word representation. *In* Alessandro Moschitti, Bo Pang et Walter Daelemans, éditeurs : *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014. URL http://aclweb.org/anthology/D/D14/D14-1162.pdf.

Catia Pesquita : Semantic similarity in the gene ontology. pages 161–173, 2017. URL https://doi.org/10.1007/978-1-4939-3743-1_12.

Jean Piaget : Le langage et la pensée chez l'enfant : Études sur la logique de l'enfant. 1923.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner et Lars Schmidt-Thieme : BPR : bayesian personalized ranking from implicit feedback. *In* Jeff A. Bilmes et Andrew Y. Ng, éditeurs : *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 452–461. AUAI Press, 2009. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25.

Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom et John Riedl : Grouplens : An open architecture for collaborative filtering of netnews. *In* John B. Smith, F. Donelson Smith et Thomas W. Malone, éditeurs : *CSCW '94, Proceedings of the Conference on Computer Supported Cooperative Work, Chapel Hill, NC, USA, October 22-26, 1994*, pages 175–186. ACM, 1994. URL http://doi.acm.org/10.1145/192844.192905.

Marco Túlio Ribeiro, Nivio Ziviani, Edleno Silva de Moura, Itamar Hata, Anísio Lacerda et Adriano Veloso : Multiobjective pareto-efficient approaches for recommender systems. *ACM TIST*, 5(4):53 :1–53 :20, 2014. URL http://doi.acm.org/10.1145/2629350.

Francesco Ricci, Lior Rokach et Bracha Shapira, éditeurs. *Recommender Systems Handbook*. Springer, 2015. URL https://doi.org/10.1007/978-1-4899-7637-6.

Norman Ricker : Wavelet functions and their polynomials. *Geophysics*, 9(3):314–323, 1944.

Stephen E. Robertson et Karen Spärck Jones : Relevance weighting of search terms. *JASIS*, 27 (3):129–146, 1976. URL https://doi.org/10.1002/asi.4630270302.

H. Rosa et D. Renault : *Accélération : Une critique sociale du temps*. Découverte-poche. Sciences humaines et sociales. La Découverte, Paris, France, 2013.

Ruslan Salakhutdinov et Andriy Mnih : Bayesian probabilistic matrix factorization using markov chain monte carlo. *In* William W. Cohen, Andrew McCallum et Sam T. Roweis, éditeurs : *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 de *ACM International Conference Proceeding Series*, pages 880–887. ACM, 2008. URL http://doi.acm.org/10.1145/1390156.1390267.

Ruslan Salakhutdinov, Andriy Mnih et Geoffrey E. Hinton : Restricted boltzmann machines for collaborative filtering. *In* Zoubin Ghahramani, éditeur : *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 de *ACM International Conference Proceeding Series*, pages 791–798. ACM, 2007. URL http://doi.acm.org/10.1145/1273496.1273596.

Matthew J. Salganik, Peter Sheridan Dodds et Duncan J. Watts : Experimental study of inequality and unpredictability in an artificial cultural market. *science*, 311(5762):854–856, 2006.

Gerard Salton : *Automatic Text Processing : The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

Rodrygo L. T. Santos, Craig Macdonald et Iadh Ounis : Exploiting query reformulations for web search result diversification. *In* Michael Rappa, Paul Jones, Juliana Freire et Soumen Chakrabarti, éditeurs : *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 881–890. ACM, 2010. URL http://doi.acm.org/10.1145/1772690.1772780.

Badrul Munir Sarwar, George Karypis, Joseph A. Konstan et John Riedl : Item-based collaborative filtering recommendation algorithms. *In* Vincent Y. Shen, Nobuo Saito, Michael R. Lyu et Mary Ellen Zurko, éditeurs : *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pages 285–295. ACM, 2001. URL http://doi.acm.org/10.1145/371920.372071.

Markus Schedl, Peter Knees, Brian McFee, Dmitry Bogdanov et Marius Kaminskas : Music recommender systems. *In* Francesco Ricci, Lior Rokach et Bracha Shapira, éditeurs : *Recommender Systems Handbook*, pages 453–492. Springer, 2015. URL https://doi.org/10.1007/978-1-4899-7637-6_13.

W. Joel Schneider et Kevin S. McGrew : The cattell-horn-carroll model of intelligence. 2012.

Barry Schwartz : *The Paradox of Choice - Why More Is Less*. Harper Perennial, New York, NY, USA, 2004.

Giovanni Semeraro, Marco De Gemmis, Pasquale Lops et Pierpaolo Basile : Combining learning and word sense disambiguation for intelligent user profiling. *In* Manuela M. Veloso, éditeur : *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2856–2861, 2007. URL http://ijcai.org/Proceedings/07/Papers/459.pdf.

Gabriel Sepulveda, Vicente Dominguez et Denis Parra : pyreclab : A software library for quick prototyping of recommender systems. *In* Domonkos Tikk et Pearl Pu, éditeurs : *Proceedings of the Poster Track of the 11th ACM Conference on Recommender Systems (RecSys 2017), Como, Italy, August 28, 2017.*, volume 1905 de *CEUR Workshop Proceedings*. CEUR-WS.org, 2017. URL http://ceur-ws.org/Vol-1905/recsys2017_poster23.pdf.

Yanir Seroussi : Utilising user texts to improve recommendations. *In* Paul De Bra, Alfred Kobsa et David N. Chin, éditeurs : *User Modeling, Adaptation, and Personalization, 18th International Conference, UMAP 2010, Big Island, HI, USA, June 20-24, 2010. Proceedings*, volume 6075 de *Lecture Notes in Computer Science*, pages 403–406. Springer, 2010. URL https://doi.org/10.1007/978-3-642-13470-8_40.

Klaus Seyerlehner, Arthur Flexer et Gerhard Widmer : On the limitations of browsing top-n recommender systems. *In* Lawrence D. Bergman, Alexander Tuzhilin, Robin D. Burke, Alexander Felfernig et Lars Schmidt-Thieme, éditeurs : *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009*, pages 321–324. ACM, 2009. URL http://doi.acm.org/10.1145/1639714.1639778.

Guy Shani et Asela Gunawardana : Evaluating recommendation systems. *In* Francesco Ricci, Lior Rokach, Bracha Shapira et Paul B. Kantor, éditeurs : *Recommender Systems Handbook*, pages 257–297. Springer, 2011. URL https://doi.org/10.1007/978-0-387-85820-3_8.

Upendra Shardanand et Pattie Maes : Social information filtering : Algorithms for automating "word of mouth". *In* Irvin R. Katz, Robert L. Mack, Linn Marks, Mary Beth Rosson et Jakob Nielsen, éditeurs : *Human Factors in Computing Systems, CHI '95 Conference Proceedings, Denver, Colorado, USA, May 7-11, 1995.*, pages 210–217. ACM/Addison-Wesley, 1995. URL http://doi.acm.org/10.1145/223904.223931.

Barry Smyth et Paul McClave : Similarity vs. diversity. *In* David W. Aha et Ian D. Watson, éditeurs : *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning, ICCBR 2001, Vancouver, BC, Canada, July 30 - August 2, 2001, Proceedings*, volume 2080 de *Lecture Notes in Computer Science*, pages 347–361. Springer, 2001. URL https://doi.org/10.1007/3-540-44593-5_25.

Harald Steck : Item popularity and recommendation accuracy. *In* Bamshad Mobasher, Robin D. Burke, Dietmar Jannach et Gediminas Adomavicius, éditeurs : *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pages 125–132. ACM, 2011. URL http://doi.acm.org/10.1145/2043932.2043957.

Harald Steck : Evaluation of recommendations : rating-prediction and ranking. *In* Qiang Yang, Irwin King, Qing Li, Pearl Pu et George Karypis, éditeurs : *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 213–220. ACM, 2013. URL http://doi.acm.org/10.1145/2507157.2507160.

Daniel Stern : Le monde interpersonnel du nourrisson. *Paris, Presses Universitaires de France*, 1989.

Ruilong Su, Li'ang Yin, Kailong Chen et Yong Yu : Set-oriented personalized ranking for diversified top-n recommendation. *In* Qiang Yang, Irwin King, Qing Li, Pearl Pu et George Karypis, éditeurs : *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 415–418. ACM, 2013. URL http://doi.acm.org/10.1145/2507157.2507207.

Alessandro Suglia, Claudio Greco, Cataldo Musto, Marco De Gemmis, Pasquale Lops et Giovanni Semeraro : A deep architecture for content-based recommendations exploiting recurrent neural networks. *In* Mária Bieliková, Eelco Herder, Federica Cena et Michel C. Desmarais, éditeurs : *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP 2017, Bratislava, Slovakia, July 09 - 12, 2017*, pages 202–211. ACM, 2017. URL http://doi.acm.org/10.1145/3079628.3079684.

Zoltán Szlávik, Wojtek Kowalczyk et Martijn C. Schut : Diversity measurement of recommender systems under different user choice models. *In* Lada A. Adamic, Ricardo A. Baeza-Yates et Scott Counts, éditeurs : *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. The AAAI Press, 2011. URL http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2817.

Gábor Takács, István Pilászy, Bottyán Németh et Domonkos Tikk : Matrix factorization and neighbor based algorithms for the netflix prize problem. *In* Pearl Pu, Derek G. Bridge, Bamshad Mobasher et Francesco Ricci, éditeurs : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 267–274. ACM, 2008. URL http://doi.acm.org/10.1145/1454008.1454049.

Gábor Takács et Domonkos Tikk : Alternating least squares for personalized ranking. *In* Padraig Cunningham, Neil J. Hurley, Ido Guy et Sarabjot Singh Anand, éditeurs : *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pages 83–90. ACM, 2012. URL http://doi.acm.org/10.1145/2365952.2365972.

Maria Taramigkou, Efthimios Bothos, Konstantinos Christidis, Dimitris Apostolou et Gregoris Mentzas : Escape the bubble : guided exploration of music preferences for serendipity and novelty. *In* Qiang Yang, Irwin King, Qing Li, Pearl Pu et George Karypis, éditeurs : *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 335–338. ACM, 2013. URL http://doi.acm.org/10.1145/2507157.2507223.

A. Toffler : *Future Shock*. Bantam Books Non-Fiction. Bantam Books, New York, New York, USA, 1971. URL https://books.google.fr/books?id=PJHi444dlRcC.

Aäron van den Oord, Sander Dieleman et Benjamin Schrauwen : Deep content-based music recommendation. *In* Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani et Kilian Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 2643–2651, 2013. URL http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.

Saúl Vargas : *Novelty and Diversity Evaluation and Enhancement in Recommender Systems*. Thèse de doctorat, Universidad AutÓnoma de Madrid, Spain, February 2015.

Saul Vargas et Pablo Castells : Rank and relevance in novelty and diversity metrics for recommender systems. *In* Bamshad Mobasher, Robin D. Burke, Dietmar Jannach et Gediminas Adomavicius, éditeurs : *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pages 109–116. ACM, 2011. URL http://doi.acm.org/10.1145/2043932.2043955.

Saul Vargas et Pablo Castells : Exploiting the diversity of user preferences for recommendation. *In* João Ferreira, João Magalhães et Pável Calado, éditeurs : *Open research Areas in Information Retrieval, OAIR '13, Lisbon, Portugal, May 15-17, 2013*, pages 129–136. ACM, 2013. URL http://dl.acm.org/citation.cfm?id=2491776.

Saul Vargas, Pablo Castells et David Vallet : Explicit relevance models in intent-oriented information retrieval diversification. *In* William R. Hersh, Jamie Callan, Yoelle Maarek et Mark Sanderson, éditeurs : *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, pages 75–84. ACM, 2012. URL http://doi.acm.org/10.1145/2348283.2348297.

Cédric Villani : Donner un sens à l'intelligence artificielle : pour une stratégie nationale et européenne. http://fichiers.acteurspublics.com/redac/pdf/2018/2018-03-28_Rapport-Villani.pdf, 2018. Mission confiée par le Premier Ministre Édouard Philippe. Mission parlementaire du 8 septembre 2017 au 8 mars 2018.

Boris Villazón-Terrazas, Luis. M. Vilches-Blázquez, Oscar Corcho et Asunción Gómez-Pérez : Methodological guidelines for publishing government linked data. pages 27–49, 2011. URL https://doi.org/10.1007/978-1-4614-1767-5_2.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio et Pierre-Antoine Manzagol : Extracting and composing robust features with denoising autoencoders. *In* William W. Cohen, Andrew McCallum et Sam T. Roweis, éditeurs : *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 de *ACM International Conference Proceeding Series*, pages 1096–1103. ACM, 2008. URL http://doi.acm.org/10.1145/1390156.1390294.

Hao Wang, Xingjian Shi et Dit-Yan Yeung : Collaborative recurrent autoencoder : Recommend while learning to fill in the blanks. *In* Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon et Roman Garnett, éditeurs : *Advances in Neural Information Processing Systems 29 : Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 415–423, 2016. URL https://tinyurl.com/y7pzlcc4.

Hao Wang, Naiyan Wang et Dit-Yan Yeung : Collaborative deep learning for recommender systems. *In* Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu et Graham Williams, éditeurs : *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1235–1244. ACM, 2015. URL http://doi.acm.org/10.1145/2783258.2783273.

Yuanyuan Wang, Stephen Chi-fai Chan et Grace Ngai : Applicability of demographic recommender system to tourist attractions : A case study on trip advisor. *In 2012 IEEE/-WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Macau, China, December 4-7, 2012*, pages 97–101. IEEE Computer Society, 2012. URL https://doi.org/10.1109/WI-IAT.2012.133.

Jacek Wasilewski et Neil Hurley : Intent-aware diversification using a constrained PLSA. *In* Shilad Sen, Werner Geyer, Jill Freyne et Pablo Castells, éditeurs : *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 39–42. ACM, 2016. URL http://doi.acm.org/10.1145/2959100.2959177.

Suyun Wei, Ning Ye, Shuo Zhang, Xia Huang et Jian Zhu : Item-based collaborative filtering recommendation algorithm combining item category with interestingness measure. *In Proceedings of the 2012 International Conference on Computer Science and Service System*,

CSSS '12, pages 2038–2041, Washington, DC, USA, 2012. IEEE Computer Society. URL http://dx.doi.org/10.1109/CSSS.2012.507.

Donald W. Winnicott : Jeu et réalité, trad. fr. *Paris, Gallimard*, 4:67, 1975.

Yao Wu, Christopher DuBois, Alice X. Zheng et Martin Ester : Collaborative denoising auto-encoders for top-n recommender systems. *In* Paul N. Bennett, Vanja Josifovski, Jennifer Neville et Filip Radlinski, éditeurs : *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, San Francisco, CA, USA, February 22-25, 2016*, pages 153–162. ACM, 2016. URL http://doi.acm.org/10.1145/2835776.2835837.

Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff G. Schneider et Jaime G. Carbonell : Temporal collaborative filtering with bayesian probabilistic tensor factorization. *In Proceedings of the SIAM International Conference on Data Mining, SDM 2010, April 29 - May 1, 2010, Columbus, Ohio, USA*, pages 211–222. SIAM, 2010. URL https://doi.org/10.1137/1.9781611972801.19.

Mi Zhang et Neil Hurley : Avoiding monotony : improving the diversity of recommendation lists. *In* Pearl Pu, Derek G. Bridge, Bamshad Mobasher et Francesco Ricci, éditeurs : *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, pages 123–130. ACM, 2008. URL http://doi.acm.org/10.1145/1454008.1454030.

Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia et Tamas Jambor : Auralist : introducing serendipity into music recommendation. *In* Eytan Adar, Jaime Teevan, Eugene Agichtein et Yoelle Maarek, éditeurs : *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*, pages 13–22. ACM, 2012. URL http://doi.acm.org/10.1145/2124295.2124300.

Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling et Yi-Cheng Zhang : Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515, 2010.

Yunhong Zhou, Dennis M. Wilkinson, Robert Schreiber et Rong Pan : Large-scale parallel collaborative filtering for the netflix prize. *In* Rudolf Fleischer et Jinhui Xu, éditeurs : *Algorithmic Aspects in Information and Management, 4th International Conference, AAIM 2008, Shanghai, China, June 23-25, 2008. Proceedings*, volume 5034 de *Lecture Notes in Computer Science*, pages 337–348. Springer, 2008. URL https://doi.org/10.1007/978-3-540-68880-8_32.

Cai-Nicolas Ziegler, Thomas Hornung, Martin Przyjaciel-Zablocki, Sven Gauss et Georg Lausen : Music recommenders based on hybrid techniques and serendipity. *Web Intelligence and Agent Systems*, 12(3):235–248, 2014. URL https://doi.org/10.3233/WIA-140294.

Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan et Georg Lausen : Improving recommendation lists through topic diversification. *In* Allan Ellis et Tatsuya Hagino, éditeurs : *Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005*, pages 22–32. ACM, 2005. URL http://doi.acm.org/10.1145/1060745.1060754.

# TABLE DES MATIÈRES

# TABLE DES FIGURES

# LISTE DES TABLEAUX