« République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

جامعة الشاذلي بن جديد
UNIVERSITE CHADLI BENDJEDID

MEMOIRE

Présenté par

REFAI AYA

Pour l'obtention de diplôme de

MASTER

Filière : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

Un système de recommandation d'hôtels basé sur les avis des clients

Soutenu le : 15/09 /2022 Devant le Jury composé de

Qualité	Nom et Prénom	Grade	Université
Président	Mr. Benmachiche A.M	MCA	**Chadli Bendjedid El-Tarf**
Rapporteur	**Mme Gasmi I.**	**MCA**	**Chadli Bendjedid El-Tarf**
Examinateur	Mme Maatallah M.	MCB	**Chadli Bendjedid El-Tarf**

Année Universitaire : 2021/2022

Remerciements

Tout d'abord, je tiens à remercier Mme GasmiIbtissem pour son soutien inconditionnel, sa disponibilité et son aide précieuse dans la poursuite de mon projet de fin d'études afin que je puisse terminer mon travail dans les meilleures conditions, J'exprime ma profonde reconnaissance et mes chaleureux remerciements aux membres de jury qui m'ont fait l'honneur de bien vouloir évaluer et juger mon travail.

Je tiens à remercier tout particulièrement ma mère pour tous ses efforts et son labeur pendant des années jusqu'à ce que j'atteigne ce que je suis actuellement, et je remercie également mon collègue et frère AthmaniHoussem El-Din pour son soutien et son aide tout au long des années.

Enfin, je tiens à exprimer ma sincère gratitude à tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

Dédicace

Je dédie ce travail:

À maman

Pour son grand amour, sa patience, ses encouragements, son sens du devoir et ses sacrifices pour moi et ses prières qui m'apportent bonheur et succès.

Pour tous mes amis qui m'ont donné du soutien et de l'énergie positive.

Tous ceux qui m'aiment et tous ceux qui l'aiment. Tous ceux que je connais de près et de loin.

Table des matières

Liste des figures

Introduction Générale

Face à la surcharge d'information, les systèmes de recommandation deviennent très utiles. Ils sont conçus pour suggérer les items les plus susceptibles d'être appréciés par les utilisateurs en fonction de nombreux facteurs différents. Ils traitent de grandes masses de données en filtrant les informations les plus importantes en fonction de plusieurs paramètres qui prennent en compte les préférences et les intérêts des usagers.

L'utilisation de techniques efficaces et robustes pour fournir des recommandations pertinentes et de bonne qualité est très utile. Les systèmes de recommandation sont généralement divisés en trois approches principales : le filtrage basé sur le contenu, le filtrage collaboratif et les méthodes hybrides. Dans les techniques de filtrage collaboratif, les degrés de similarité sont calculés pour trouver le voisinage de l'utilisateur ou de l'objet. En effet, il existe plusieurs méthodes pour calculer la similarité, notamment la similarité de cosinus, le coefficient de corrélation de Pearson, la similarité de Jaccard, etc. (Jain et al., 2020)[w1].Cependant, lorsque le système souffre de la rareté des données, l'utilisation des mesures de similarité ci-dessus devient difficile car le nombre des évaluations communes entre les utilisateurs est très faible. Par conséquent, nous utilisons, dans ce travail, la méthode LDA pour identifier les propriétés latentes entre les hôtels à partir des avis des utilisateurs. Les degrés de similarité sont ensuite calculés dans cet espace latent pour identifier le voisinage et faire ainsi des recommandations pertinentes.

Ce mémoire est organisé en trois chapitres. Le premier chapitre est consacré à la présentation des concepts de base sur les systèmes de recommandation et les techniques de recommandation.

Le deuxième chapitre est dédié à une étude conceptuelle détaillée du système de recommandation proposé.

Le dernier chapitre présente les outils et les langages utilisés pour le développement de notre modèle ainsi que les différents résultats obtenus.

Enfin, nous terminons ce mémoire par une conclusion générale et quelques perspectives.

Chapitre 1 : Etat de l'Art

1. Introduction

Au cours des dernières décennies, de nombreux efforts de recherche ont été déployés pour essayer d'utiliser de nouvelles méthodes pour remédier à bon nombre des limites des systèmes de recommandation, telles que l'amélioration de la qualité des recommandations. Nous aborderons ici les fondements théoriques de notre travail. Qui est divisé en trois sections. La première partie présente les systèmes de recommandation et leurs différentes techniques. La partie 2 décrit la structure et les techniques d'évaluation des clients. La partie 3 présente l'intégration de plusieurs travaux liés à l'application de la méthode de l'avis client dans le cadre de recommandations.

2. Historique:

Première génération : basée sur l'algorithme "BetweenProducts" : les produits sont liés par leurs propriétés. Ce type de système de recommandation est apparu dans le Web 1.0 et les utilisateurs étaient strictement limités aux rôles documentaires. Deuxième Génération : Basé sur un filtrage collaboratif qui propose des produits d'autres consommateurs déclarés. La logique recommandée est "entre utilisateurs", qui relie les utilisateurs en fonction de leur profil et de leurs paramètres. Les données collectées seront utilisées pour reconstruire la navigation des consommateurs et créer des profils de consommateurs [w2].Il est également important de penser à toutes les informations déclaratives que l'utilisateur saisit, c'est-à-dire la carte / le formulaire qui est saisi sur la page Web. Ce type de système de recommandation est venu avec le web 2.0, apportant avec lui une nouvelle culture sociale : des profils de consommateurs reconstruits à partir de profils similaires. Troisième génération : des contenus basés sur un filtrage hybride qui surpasse les

deux générations précédentes + les innovations des NTIC collaboratives dans la société conduisent à une nouvelle conception et de nouveaux usages des agents de recommandation, avec une augmentation du stockage, de nouveaux appareils abordables et de nouveaux traitements de données. Ses racines se trouvent notamment dans le mouvement Big Data[w3].

Les agents de troisième génération peuvent modéliser les préférences des consommateurs, les préférences des consommateurs, sans intervention directe en les formulant explicitement. A priori, ils ne s'appuient sur aucune définition formelle des préférences, mais passent par des théories d'apprentissage, c'est-à-dire que les consommateurs recherchent des outils pour pouvoir définir leurs préférences. Au fur et à mesure que Jean-Sébastien Vayre a pu grandir, l'objectif principal de ces dispositifs n'était pas de comprendre le comportement des utilisateurs mais d'automatiser au maximum leurs demandes et leurs résultats en prévoyant d'anticiper leurs besoins, avec leur volonté commerciale en jeu (incitation à la consommation)[w4].

3. Les systèmes de recommandation

Définitions :

Les systèmes de recommandation (RS) sont une forme spécifique de filtrage de données qui vise à présenter des informations susceptibles d'intéresser les utilisateurs. Ils peuvent être définis comme des programmes qui cherchent à recommander l'article (produit ou service) le plus approprié à un utilisateur particulier (individu ou entreprise) en déterminant l'intérêt de l'utilisateur pour l'article en fonction des informations sur l'article. L'interaction entre l'utilisateur et eux les prédit. (Bobadilla et al., 2013) [1].

L'objectif du développement de systèmes de recommandation est de réduire la surcharge d'informations en extrayant les informations et les services les plus pertinents à partir de grandes quantités de données et en fournissant des services personnalisés. (Resnick et Varian, 1997) [2]. Les systèmes de recommandation sont définis de différentes manières. La définition la plus courante et la plus courante citée ici est celle de Robin Burke (Burke, 2002) [3].

Le système de recommandation est le suivant. Un système qui peut fournir des recommandations personnalisées, générales ou d'utilisateurs dans de grands espaces pour guider les ressources utiles. Dans tout système de recommandation, deux entités forment le centre du système. Toutes les techniques recommandées tournent autour de ces entités : utilisateurs et articles. Un utilisateur est une personne qui interagit avec le système, encourage le système à recommander un article, puis commente l'article. Les articles sont un terme général utilisé pour désigner les ressources que le système fournit aux utilisateurs. Outre les utilisateurs et les articles, l'espace d'informations du système de recommandation contient également des paramètres d'articles représentés par les utilisateurs, souvent appelés évaluations (utilisateurs, articles, évaluations). Ces notes peuvent prendre plusieurs formes. Cependant, la plupart des systèmes utilisent des évaluations sous forme d'échelles de 1 à 5 ou d'évaluations binaires. Un ensemble de triplets (utilisateurs, éléments, notes) forme ce que l'on appelle le bloc-notes. Les paires (utilisateurs ; éléments) pour lesquelles l'utilisateur n'a pas évalué l'élément sont des valeurs inconnues dans le tableau.

4. Technique de recommandation

4.1. Filtrage basé sur le contenu

Il s'agit d'un type particulier des systèmes de recommandation qui se base sur le principe de contenus similaires. Ainsi, le filtrage basé sur le contenu utilise des descriptions similaires entre les items à recommander à l'utilisateur actuel et les items que cet utilisateur a appréciés dans les recommandations précédentes. Par exemple, si un utilisateur lit un livre, le système proposera d'autres livres de même contenus ou de même genres que ce livre.

L'avantage de ce système est qu'il peut affecter des documents à des profils utilisateurs. Les utilisateurs ne sont pas conditionnés par d'autres utilisateurs, ils peuvent donc recevoir des recommandations même lorsqu'ils utilisent le système seul, évitant certaines limitations et lacunes des systèmes de collaboration basés sur le contenu (Belloui, 2008) [4].

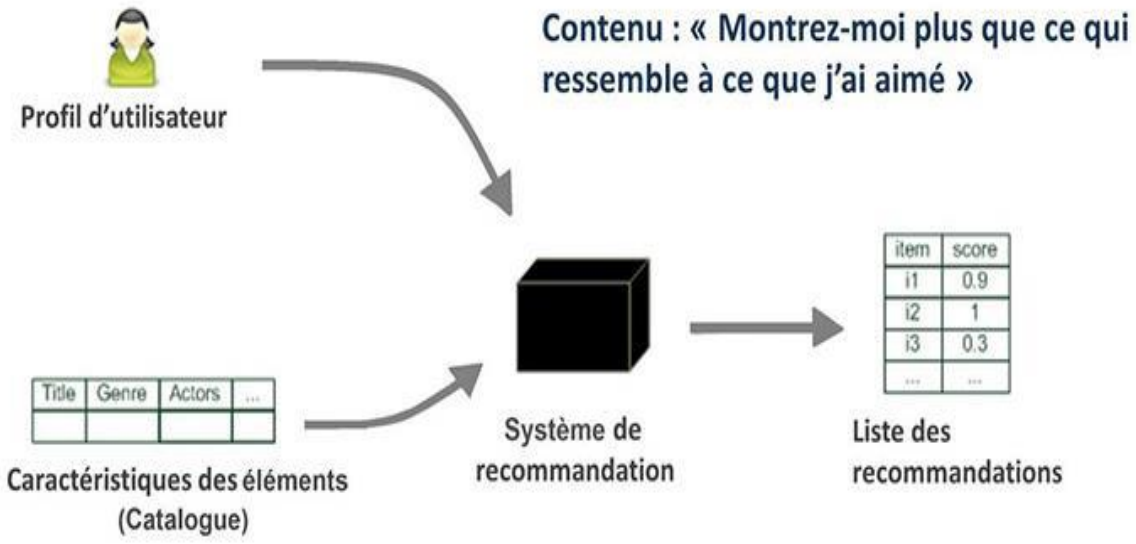

Figure 1.4.1: *Un système de recommandation basé sur le contenu* [***1.4.1***]

4.2. Filtrage collaboratif

Le filtrage collaboratif se base sur la matrice user×item contenant les évaluations des utilisateurs sur les items. Il fait correspondre les utilisateurs ayant des préférences similaires en calculant les degrés de corrélation entre leurs profils pour faire des recommandations. Les utilisateurs similaires forment un groupe appelé voisinage. Ainsi, le système propose à l'utilisateur des objets qu'il n'a pas encore évalués à condition qu'ils aient été déjà appréciés par les utilisateurs de son voisinage.

Ce type de recommandation ne considère pas le contenu des items. Il peut donc être appliqué à n'importe quel type de donnée. Les techniques de filtrage collaboratif sont très populaires, mais elles présentent de nombreux inconvénients, notamment ceux liés au manque de données. Si la matrice des évaluations est creuse, il sera difficile d'identifier les plus proches voisins et par conséquent les recommandations ne seront plus pertinentes.

Le filtrage collaboratif souffre également du problème de démarrage à froid. Il ne peut pas sélectionner les utilisateurs intéressés par un nouvel item sauf si cet item est noté par un certain nombre d'utilisateurs. Cette limitation concerne également les nouveaux utilisateurs. Ainsi, ces derniers doivent faire quelques évaluations afin qu'ils soient satisfaits des recommandations qu'ils reçoivent. Il existe deux types de filtrage collaboratif : le filtrage basé sur la mémoire et le filtrage basé sur le modèle (Goldberg K., Roeder T., Gupta D., Perkins C.2001) [5] .

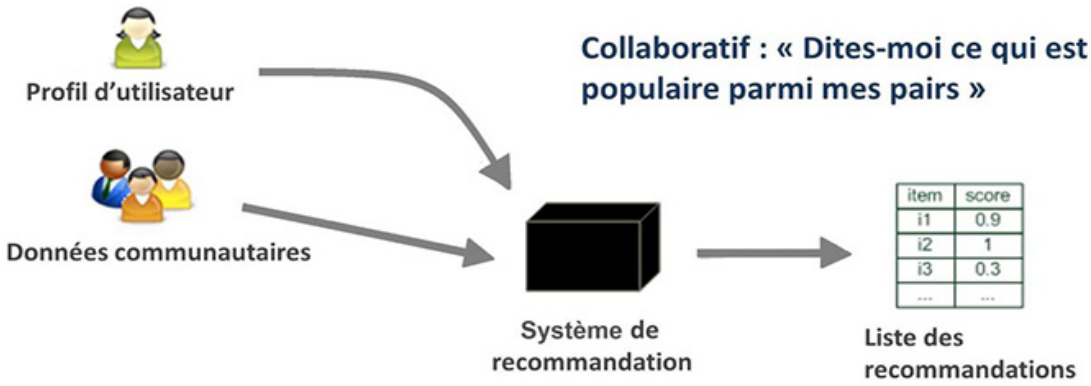

Figure 1.4.2 : *Un système de recommandation collaboratif* [***1.4.2***]

4.3. Filtrage hybride

Les systèmes de filtrage hybrides est un combinaison du système de filtrage collaboratif avec des systèmes basés sur le contenu pour faire des divinations ou des recommandations pour éviter certaines limitations et inconvénients des systèmes basés sur le contenu et collaboratifs (Rao &Talwar, 2008) [6].

Généralement, le filtrage hybride utilise des méthodes pour accorder les ensembles de recommandations telles que la pondération, la cascade, la commutation, etc. afin de produire des recommandations finales pour les utilisateurs (Burke, 2002) [3].

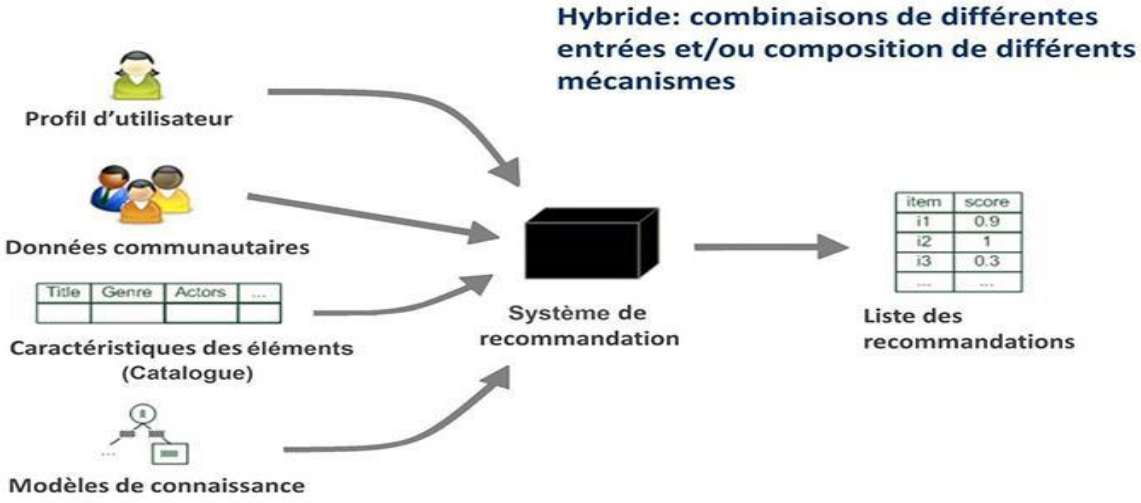

Figure 1.4.3 : *Le système de recommandation hybride* [***1.4.1***]

5. Système de recommandation basé LDA :

5.1. Latent Dirichlet Allocation (LDA)

L'allocation latente de Dirichlet (LDA) est une méthode probabiliste générative permettant d'extraire des sujets latents à partir de corpus de textes. Elle associe un texte, considéré comme une liste de mots non ordonnée, à un vecteur de thèmes (Blei et al., 2003) [7]. LDA modélise chaque document comme un mélange aléatoire de sujets latents et définit chaque sujet par une distribution de mots. Elle génère les distributions de probabilité des sujets p(t|d) pour chaque document d. Chaque sujet t (t ∈{1, . . . , T }) est composé des probabilités de mots p(wj|t) pour les mots wj, j = 1, . . . , V. Où V est la taille du vocabulaire et T est le nombre prédéfini de thèmes.

La probabilité des sujets dans le modèle est contrôlée par les hyperparamètres de Dirichlet α et β, qui jouent un rôle important dans l'apprentissage de thèmes latents. α est le paramètre de concentration a priori de Dirichlet qui représente la densité des sujets dans les documents. β est le paramètre de concentration a priori qui contrôle la distribution des mots par sujet (Al-Ghossein et al., 2018) [W5].

5.2Quelques travaux de recherche à base de LDA en système de recommandation

Les propriétés latentes peuvent être très utiles pour augmenter l'efficacité des systèmes de recommandation. Par conséquent, des outils non supervisés tels que LDA ont été utilisés pour apprendre les caractéristiques latentes dans les systèmes de recommandation (Al-Ghossein et al., 2018) [W5].

Yan et al (Yan et al., 2016) [8]ont présenté un modèle de recommandation de vidéos à base de LDA pour Youtube. Ils ont intégré dans leur modèle des informations sur le contenu des vidéos et des informations sociales sur les utilisateurs, extraites à partir du réseau Twitter.

(Bagul et Barve) [9]Bagul et Barve ont proposé un système de recommandation basé sur le contenu en utilisant l'allocation de Dirichlet latente afin de générer des recommandations pour les publications scientifiques. Le système suggère des revues et des conférences appropriées pour publier un travail de recherche en se basant sur le résumé de l'article scientifique.

Lin et al. (Lin et al., 2017) [10] ont utilisé les avis des utilisateurs de la plateforme Airbnb pour déduire les caractéristiques des produits afin d'apprendre les préférences des clients. Les auteurs ont mis en œuvre la technique LDA pour extraire à la fois les caractéristiques et les préférences. Le modèle a été réalisé sur les avis des consommateurs de tous les produits. Chaque produit a été défini en fonction de la distribution des probabilités des thèmes latents.

Herwanto et Ningtyas(Herwanto and Ningtyas) [11] ont proposé un système de recommandation en se basant sur l'exploration de l'utilisation du Web et la modélisation de thèmes latents. Ils ont appliqué la méthode LDA sur le contenu extrait à partir du site Web. Al-Ghossein et al. (Al-Ghossein et al., 2018) [W5]ont proposé une approche adaptative de modélisation collaborative des thèmes latents pour la recommandation en ligne. Ils ont combiné AWILDA, une version adaptative de LDA qui est capable d'analyser et de modéliser le flux des documents. Ils ont également appliqué la factorisation matricielle incrémentielle pour connaître les préférences des utilisateurs en fonction de leurs interactions.

6. Limitation des approches basé sur le contenu :

La principale limitation de la recommandation basée sur le contenu est qu'elle nécessite de collecter un nombre suffisant d'attributs décrivant la ressource. C'est pourquoi elle est appropriée dans le cadre de ressources textuelles ou lorsque des descriptions textuelles de sources ont été saisies manuellement. Dans le cadre des sources textuelles, une des limites vient des méthodes de classement des textes utilisées : en effet, deux ressources peuvent être similaires du point de vue de leurs propriétés, mais d'une qualité ou d'une pertinence incomparable.

Une autre limitation est que ces modèles ne peuvent recommander que des ressources similaires à celles qu'un utilisateur donné aime, ce qui empêche de recommander d'autres ressources que le même utilisateur peut également recommander. Pour pallier ce problème, il est possible de faire des recommandations au hasard parmi les recommandations.

Enfin, une dernière limitation est qu'un nouvel utilisateur d'un tel système doit consulter ou noter certaines ressources avant que le système puisse lui fournir des recommandations pertinentes. Ce problème est appelé démarrage à froid.[W6]

Une façon de pallier ce problème est de demander un certain nombre d'informations à l'utilisateur lors de son arrivée et d'utiliser un profil standard qui correspond aux informations qu'il a fournies.

7. Métriques d'évaluation des systèmes de recommandation :

Mesurer la qualité d'un sujet est une difficulté bien connue. La confusion a été utilisée dans les évaluations traditionnelles, mais il a été démontré qu'elle était négativement corrélée à l'interopérabilité mesurée par les humains du sujet en envahissant le sujet à l'aide de nouveaux mots et de nouvelles tâches .Depuis, plusieurs autres méthodes ont été proposées pour évaluer automatiquement la qualité des sujets. Newmann, etc. montre que le pointwisemutual information (PMI) est fortement corrélé à l'évaluation humaine de la cohérence sémantique. Le PMI utilise des données externes (souvent de Wikipédia en anglais) pour mesurer les associations de mots entre des paires de mots dans un sujet. Minno et al proposent une métrique de cohérence du sujet qui mesure les paroles du sujet.

Cooccurrence de documents pour identifier les sujets de mauvaise qualité et montrer qu'ils sont en corrélation avec des notes sur des sujets d'experts. Cependant, des travaux antérieurs ont montré que l'utilisation de mesures individuelles pour évaluer la qualité d'un sujet est problématique. En effet, différentes mesures capturent généralement différents aspects de la qualité. Par exemple, mesurer si un sujet représente une idée cohérente et facile à lire, mesurer l'éventail des sujets qui existent réellement dans le corpus ou, comme dans notre cas, la substance du sujet final. mots cibles et liés au domaine. .. Dans cette tâche, nous nous concentrerons sur le problème des mots vides, qui est un obstacle à la modélisation, car les mots vides dominent la fréquence des mots et les statistiques de cooccurrence du corpus. Dans de nombreux modèles LDA, les rubriques représentent principalement des mots

courants qui masquent le contenu pertinent du corpus. De plus, en présence de mots vides, nous constatons que les métriques LDA conçues pour évaluer d'autres aspects de la qualité du sujet fonctionnent de manière contre-intuitive.[W7]

8. Les systèmes de recommandation basés sur le contenu présentent les avantages et les inconvénients suivants :

8.1. Les avantage :

- ils recommandent des éléments similaires à ceux que les utilisateurs ont aimés dans le passé.
- ils prennent en compte le profil des utilisateurs qui est la clé pour avoir les recommandations les plus pertinentes pour chacun.
- faire coïncider les préférences de l'utilisateur et les caractéristiques des éléments fonctionne pour de nombreux types de données (textuelles, numériques, etc.) puisqu'on utilise généralement des listes de mots-clés.
- les données relatives aux autres utilisateurs sont inutiles.
- il n'y a pas de problème de démarrage à froid lorsqu'un nouvel élément est ajouté au catalogue ou de faible densité puisqu'il s'agit de faire coïncider les préférences de l'utilisateur et les caractéristiques des éléments.
- il est possible de faire des recommandations à des utilisateurs avec des goûts « uniques ».
- il est possible de recommander de nouveaux éléments ou même des éléments qui ne sont pas populaires.

8.2. Les inconvénients :

- tous les contenus ne peuvent pas être représentés avec des mots-clés (par exemple, les images).
- des éléments représentés par le même ensemble de mots-clés ne peuvent pas être distingués.
- les utilisateurs ayant visualisé un très grand nombre d'éléments posent un problème (trop d'informations dans le profil de l'utilisateur à faire coïncider avec les caractéristiques des éléments).
- lorsqu'un nouvel utilisateur commence à utiliser le système, il n'existe pas d'historique.
- un risque de « sur-spécialisation » apparaît, c'est-à-dire que l'on se limite aux éléments similaires et que les réponses sont trop homogènes.
- les profils des utilisateurs restent difficiles à élaborer et, qui plus est, il faut prendre en compte l'évolution des intérêts de l'utilisateur.
- pour que le système produise des recommandations précises, l'utilisateur doit fournir un *feedback* sur les suggestions retournées mais cela est chronophage pour lui.
- finalement, ces systèmes sont entièrement basés sur les scores d'éléments et les scores d'intérêt : moins il y a de scores, plus l'ensemble de recommandations possibles est limité.

9. Conclusion :

Dans le chapitre 1 on fait un aperçu sur les systèmes de recommandation .nous abordons l'histoire et les principaux types de systèmes de recommandation : ceux basés sur le contenu, ceux qui s'appuient sur le filtrage collaboratif, et enfin ceux qui s'appuient sur les approches hybrides. Les avantages et les inconvénients de Filtrage basé sur le contenu et ces limitations et les métriques d'évaluation et on fait une petite description sur la méthode de Latent Dirichlet Allocation (LDA).

L'un des principaux inconvénients est le fait que ces systèmes sont toujours finis et ne peuvent pas s'adapter à l'environnement dans lequel ils se trouvent, ou en d'autres termes, c'est l'ensemble des éléments qui influencent la compréhension d'une situation donnée. Pour cette raison, des systèmes de recommandation ont vu le jour, et leur qualité est étroitement liée à la capacité à prendre en compte le contexte dans lequel se trouve l'utilisateur quand il veut une recommandation. Nous expliquerons en détail dans le chapitre 2 .

Chapitre 2 : Etude Conceptuelle

1. Introduction

Ce chapitre décrit les détails conceptuels du système de recommandation que nous avons proposé. Nous commençons par énoncer notre problématique et les objectifs de notre travail. Nous présentons ensuite une description détaillée de l'architecture générale de notre modèle.

2. Motivation et objectif

La recommandation basée sur le contenu consiste à analyser le contenu des items candidats à la recommandation ou les descriptions de ces items. Les méthodes de recommandation basées sur le contenu utilisent des techniques largement inspirées du domaine de la recherche d'information. La différence se trouve essentiellement dans l'absence de requêtes explicites formulées par l'utilisateur. Les approches basées contenu infèrent plutôt les préférences de l'utilisateur et lui recommandent les items dont le contenu est similaire au contenu des items qu'il a aimés auparavant.

Ainsi, quand de nouveaux items sont introduits dans le système, il peuvent être recommandés directement, sans que cela ne nécessite un temps d'intégration comme c'est le cas pour les systèmes de recommandation basés sur une approche de filtrage collaboratif

Pour résoudre ce problème, le modèle proposé dans le cadre de ce travail exploite les propriétés cachées des items à l'aide de la méthode LDA. Ces propriétés latentes sont utilisées pour calculer la matrice de similarité qui sera utilisée dans le calcul des prédictions afin de suggérer des hôtels appropriés à l'utilisateur.

3. Problématique

Comment appliquer la méthode LDA sur un system de recommandation basé sur les Review des clients?

4. Architecture générale du système

L'architecture générale du système de recommandation proposé est illustrée par la figure suivante.

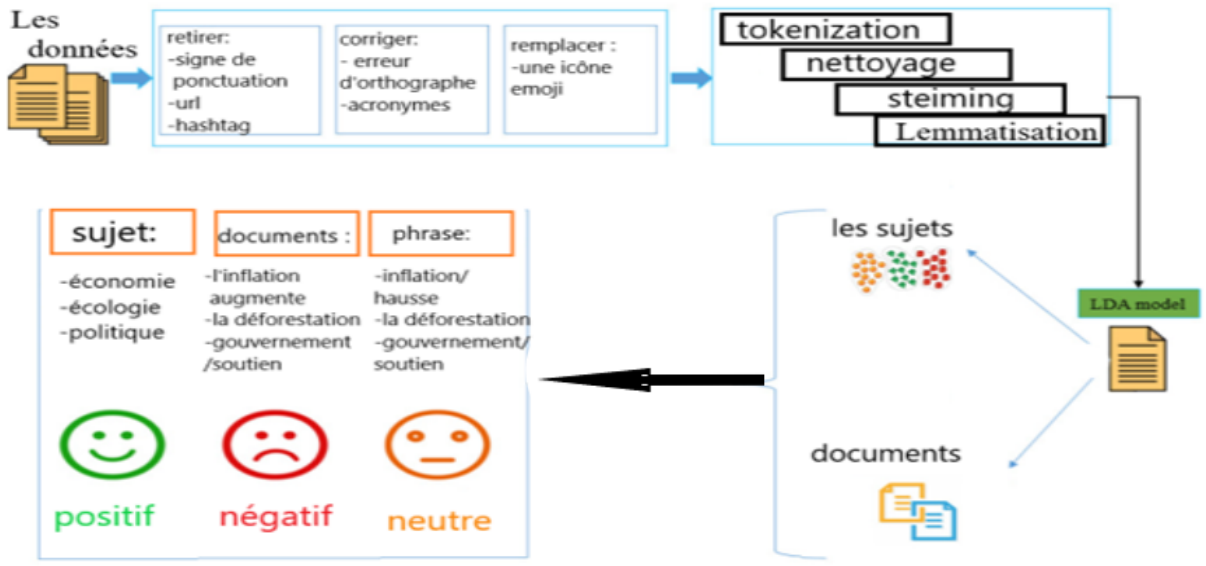

Figure 2.4 : L'architecture du modèle proposé

Le problème de la classification des sentiments est relativement plus difficile que la classification traditionnelle basée sur les sujets. En effet, les émotions peuvent être exprimées de manière plus nuancée, tandis que les thèmes peuvent être plus facilement identifiés en tenant compte des cooccurrences de mots-clés. Selon le groupe d'évaluation, l'amélioration de la précision de la détection de la polarité de l'humeur est associée à l'inclusion d'informations préalables ou de dictionnaires subjectifs. Dans cette étude, nous proposons la méthode LDA pour déterminer la polarité de l'humeur d'abord au niveau du sujet, puis au niveau du document et du mot. Nous avons appliqué le modèle de sélection de fonctionnalités LDA pour découvrir des thèmes dans l'ensemble de données TripAdvisor. Pour visualiser les distributions thématiques, nous avons utilisé l'outil de visualisation de données LDA vis développé par Carson et al. Aspects divulgués des relations sujet-concept, y compris la distance du sujet, le nombre de grappes, le concept et les calculs de valeur lambda. Lambda (_) est utilisé pour calculer le poids accordé à la probabilité des termes d'un sujet par rapport à leur plage (mesurées sur une échelle logarithmique) et les termes les plus pertinents pour chaque sujet. Détermine la valeur du lambda Le processus global de cette recherche est illustré à la figure *2.4*.

5. Base de données utilisée

Pour analyser le contenu textuel, un jeu de données est nécessaire. L'ensemble de données doit contenir du texte et des évaluations afin qu'ils puissent être annotés. Pour répondre à ces critères, TripAdvisor a été sélectionné comme source de données. TripAdvisor, fondé en 2000, est un voyage et un restaurant américain société de site Web qui affiche des critiques d'hôtels et de restaurants, des réservations d'hébergement et d'autres contenus liés aux voyages. Il aurait pu y avoir d'autres sélections de données sources telles que les avis Google-Places. Cependant, TripAdvisor est un site Web populaire en Grèce, qui est un site Web vieux de 19 ans et qui a plus de contenu que GooglePlaces. TripAdvisor ne fournit pas de données ou d'accès API, donc un Web-Scraper devait être créé afin de construire un jeu de données

APPLICATION DE LAMETHODE LDA

6. Préparation des données

Dans la première phase, les commentaires des clients sont prétraités par la boîte à outils python NLTK pour filtrer les données bruyantes et les mots non informatifs qui n'ajoutent aucun caractère distinctif dans les textes. En particulier, chaque phrase est segmentée en une liste de mots. Les mots vides, les ponctuations, etc. sont supprimés. Les majuscules sont converties en minuscules. Enfin, les termes issus d'un même lemme ayant des différences dues aux morphologies flexionnelles sont unifiés (Lemmatisation).

Tous les mots prétraités sont extraits pour construire le vocabulaire. Ainsi, chaque commentaire est considéré comme un sac de mots (BoW). Il est représenté sous forme de vecteur dans cet espace de termes, avec une valeur dans chaque cellule indiquant le nombre d'occurrences du mot correspondant.

6.1. Tokenisation

Au cours de cette étape, le texte est décomposé en unités élémentaires. Dans la plupart des cas, les mots sont séparés par la présence d'espaces ou de signes de ponctuation. Cependant, le choix de la règle de tokenisation a un impact significatif sur les performances et nécessite une connaissance préalable des données traitées .

6.2. Nettoyage

Le nettoyage de données est l'opération de détection, correction et suppression d'erreurs présentes dans la basede données. Les erreurs peuvent être des erreurs de frappe, des informations manquantes, etc.

6.3. Stemming ou raconisation

Stemming est le processus de réduction des mots à leur forme de racine. Cela consiste à ne conserver que la racine des mots étudiés. L'idée étant de supprimer les suffixes, préfixes des mots afin de ne conserver que leur origine. Par exemple, le mot ***prétraitement***est

remplacé par***traite***. Les mots ***économie***, ***économiquement***, ***économiste***sont remplacé par***économ***.

6.4. Lemmatisation

Dans cette étape les mots d'une même famille appartenant à un texte sont réduits en une unique entité que l'on appelle un ***lemme*** . Elle consiste donc à représenter les mots sous leur forme canonique. Par exemple un verbe est remplacé par son infinitif et un nom par son masculin singulier. L'idée consiste àne conserver que le sens des mots utilisés dans le corpus.

7. Dictionnaire de vocabulaire

Nous avons utilisé la bibliothèque Gensim fournie par Python pour effectuer le sujet la modélisation. Les étapes de création du modèle de sujet sont décrites ci-dessous :

- Créer un dictionnaire :Un dictionnaire est créé à partir d'un sac de mots dans l'ensemble de données. De plus, le dictionnaire montre combien de mots et comment plusieurs fois l'apparition de ces mots dans l'ensemble de données.
- Construire un corpus : Le dictionnaire s'est ensuite transformé en un corpus TF-IDF (Term FrequencyInverse Document Frequency) en convertissant une liste document dans un format de matrice de termes de document.
- Construire le modèle LDA et le calcul de la valeur de cohérence : Dans cette recherche, nous construisons un modèle LDA à l'aide d'un corpus et d'un dictionnaire, et le nombre de sujets comme paramètre d'entrée. Le corpus et les paramètres du dictionnaire sont obtenus à partir des tâches précédentes.

De plus, nous formons le modèle en attribuant 1 à 10 nombre de sujets et en calculant la valeur de cohérence pour chaque nombre de topic pour obtenir le meilleur nombre de topic.

8. extraction des thèmes

L'étape de prétraitement est suivie de l'apprentissage d'un modèle à base de LDA pour apprendre les propriétés cachées en termes de distributions de probabilité des thèmes dans chaque commentaire et les distributions de probabilité des mots sur chaque thème. LDA génère les distributions de probabilité des sujets p(T|I) pour chaque commentaire I, où les sujets T = (T1, T2, . . . ,Tn_sujets) sont composés des probabilités de mots p(Wj|Ti) pour les mots Wj, j = 1, . . . , n_mots où n_mots est le nombre de mots du vocabulaire. Notez que n_topics est le nombre prédéfini de sujets. Les sujets sont considérés comme les caractéristiques cachées des commentaires, et la densité de sujets dans le modèle est contrôlée par les hyperparamètres de Dirichlet α et β qui jouent un rôle important dans l'apprentissage du modèle proposé.

9. Calcul de similarité

Chaque commentaire est représenté dans l'espace des sujets latents en utilisant les distributions de de probabilité commentaire-thème comme vecteur de caractéristiques. La similarité entre les items i

etj dans l'espace des sujets latents est calculée en utilisant le cosinus entre leurs distributions de probabilité. Cette mesure est exprimée par l'équation suivante :

$$\mathrm{sim}(i,j) = \cos(\overrightarrow{x_i}, \overrightarrow{x_k}) = \frac{\sum_{i \epsilon U_{ij}} r_{u,i} * r_{u,j}}{\sqrt{\sum_{i \epsilon U_{ij}} r^2{}_{u,i}} \cdot \sqrt{\sum_{i \epsilon I_{ij}} r^2{}_{u,j}}} \qquad (2.9)$$

10. Calcul de prédiction

L'objectif de cette étape est de prédire les préférences de l'utilisateur courant sur les hôtels en utilisant les résultats obtenus à partir de l'étape précédente, la prédiction est donc calculée selon la formule suivante :

$$\text{pred(u, i)} = \frac{\sum_{w \epsilon voissins(u) \frown U_i} r_{w,i}}{|voissins(u) \frown U_i|} (2.10)$$

11. Calcul de l'erreur

La métrique utilisée pour l'évaluation de l'algorithme proposé est la racine de l'erreur moyenne quadratique (MeanSquardError : RMSE) calculer selon l'équation suivante:

$$e_G^{relatif} = \frac{G_{mes} - G_{ref}}{G_{ref}} * 100 \quad (2.11)$$

12. Conclusion

Dans ce chapitre, on a discuté de la méthode de travail et des algorithmes. On a proposé un schéma global du système de recommandation souhaité dont chaque étape est détaillée d'une façon théorique rien que pour montrer les fonctions, les formules et les outils de développement utilisés. La pratique était basée sur la théorie qu'on a déjà expliquée. Dans le chapitre de suite, on va tester la qualité du système pour obtenir les meilleurs résultats.

Chapitre 3 : État de la technique

1. Introduction

Dans cette section, nous décrivons le travail d'application et les détails de la mise en œuvre effectuée au cours de ce travail. Nous allons d'abord définir la base de données utilisée pour valider notre approche. Ensuite, nous expliquons les outils et le langage d'application. Nous détaillons ensuite la mise en œuvre des étapes de notre travail et concluons par les résultats et expérimentations.

2. Environnement matériel

L'algorithme décrit dans ce chapitre est développé sur un Pc équipé d'un Intel Core i5 - 5300U 2.0Ghz (5e Génération) et 8 Go de RAM. Il est développé en langage python sous l'environnement de développement Jupyter.

3. Environnement logiciel

Pour déévelopper notre application, nous avons utilisés :

Python version 3.7 : Python est un langage de programmation qui peut s'utiliser dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à des bibliothèques spécialisées. Il est cependant particulièrement utilisé comme langage de script pour automatiser des tâches simples mais fastidieuses, comme un script qui récupérerait la météo sur Internet ou qui s'intégrerait dans un logiciel de conception assistée par ordinateur afin d'automatiser certains enchaînements d'actions répétitives (voir la section Adoption). On l'utilise également comme langage de développement de prototype lorsqu'on a besoin d'une application fonctionnelle avant de l'optimiser avec un langage de plus bas niveau. Il est particulièrement répandu dans le monde scientifique, et possède de nombreuses bibliothèques optimisées destinées au calcul numérique

Jupyter notebook: Jupyter est une application web utilisée pour programmer dans plus de 40 langages de programmation, dont Python, Julia, Ruby, R, ou encore

Scala2. C'est un projet communautaire dont l'objectif est de développer des logiciels libres, des formats ouverts et des services pour l'informatique interactive. Jupyter est une évolution du projet IPython. Jupyter permet de réaliser des calepins ou notebooks, c'est-à-dire des programmes contenant à la fois du texte en markdown et du code. Ces calepins sont utilisés en science des données pour explorer et analyser des données.

Bibliothèque et Framework utilisé :

Pandas : Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.

Pandas est un logiciel libre sous licence BSD2. Son nom est dérivé du terme Panel Data (en français "données de panel", un terme d'économétrie pour les jeux de données qui comprennent des observations sur plusieurs périodes de temps pour les mêmes individus). Son nom est également un jeu de mots sur l'expression "Python Data Analysis".

Matplotlib : Matplotlib est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques5. Elle peut être combinée avec les bibliothèques python de calcul scientifique NumPy et SciPy6. Elle fournit également une API orientée objet, permettant d'intégrer des graphiques dans des applications, utilisant des outils d'interface graphique polyvalents tels que Tkinter, wxPython, Qt ou GTK.

Matplotlib est distribuée librement et gratuitement sous une licence de style BSD4. Sa version stable actuelle (la 2.0.1 en 2017, la 3.5.0 en novembre 2021) est compatible avec la version 3 de Python.

Wordcloud : Lewordcloud ou nuage de mots-clés, ou nuage de tags (en anglais wordcloud ou keyword cloud) est une représentation visuelle des mots-clés (tags) les plus utilisés sur un site web ou une base de données. Généralement, les mots s'affichent dans des tailles et graisses de caractères d'autant plus visibles qu'ils sont utilisés ou populaires.

Scikit-learn : Scikit-learn est une bibliothèque libre Python destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeursnotamment dans le monde académique par des instituts français d'enseignement supérieur et de recherche comme Inria.

Elle propose dans son framework de nombreuses bibliothèques d'algorithmes à implémenter, clé en main. Ces bibliothèques sont à disposition notamment des data scientists.

Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s'harmoniser avec d'autres bibliothèques libres Python, notamment NumPy et SciPy

4. Méthodologie d'implémentation

Description de l'application :

Nous allons présenter les interfaces principales de notre application et les méthodes utilisé

1) importation

```
[ ]  import numpy as np
     import pandas as pd

     import matplotlib.pyplot as plt

     import nltk
     from nltk.stem import WordNetLemmatizer

     import gensim
     from gensim.models.coherencemodel import CoherenceModel
     from gensim.models.ldamodel import LdaModel
```

Figure 3.4.1 : Implémentation

2) Préparation des données

```
[ ] data = pd.read_csv("TripAdvisor.csv", encoding = "latin")
```

```
[ ] data.head(5)
```

	id	name	review_id	date	rating	user	user_add	user_contibution	helpful_votes	stayed	travelled_as	review_title	review_text	Unnamed: 13	Unnamed: 14
0	883723092	A Victory Inn Tolleson	546661306	12/11/2017	2	txhcmtg	Kerrville	63	33	NaN	NaN	Very Sketchy	I was traveling between California and Texas; ...	NaN	NaN
1	883723092	A Victory Inn Tolleson	524211843	9/13/2017	1	Sean G	Las Vegas; Nevada	7	9	NaN	NaN	My car was broken into! Pimps; thugs and prost...	My car was broken into (See pictures)... pumps...	NaN	NaN
2	883723092	A Victory Inn Tolleson	500419232	7/10/2017	1	vr4dad	NaN	1	NaN	17-Jul	traveled solo	Roaches in room 102 can't get a hold of manage...	I check in to hotel and was in a room across p...	NaN	NaN
3	883723092	A Victory Inn Tolleson	460995068	2/18/2017	1	Kate L	NaN	1	NaN	NaN	NaN	Horrific	The most disgusting motel rooms I have ever ex...	NaN	NaN
4	883723092	A Victory Inn Tolleson	431610872	10/25/2016	5	IJEAN B	NaN	1	1	16-Oct	traveled with family	Comfortable and convenient	After traveling for 2 day's from Oregon to Ari...	NaN	NaN

Figure 3.4.2 :Préparation des données

3) Suppression des données supplémentaires

```
[ ] data = data[[col for col in data.columns if col not in ['date','rating','user','user_add','user_contibution','helpful_votes','stayed','travelled_as','review_title','Unnamed: 13','Unnamed: 14']]]
    data.head()
```

	id	name	review_id	review_text
0	883723092	A Victory Inn Tolleson	546661306	I was traveling between California and Texas; ...
1	883723092	A Victory Inn Tolleson	524211843	My car was broken into (See pictures)... pumps...
2	883723092	A Victory Inn Tolleson	500419232	I check in to hotel and was in a room across p...
3	883723092	A Victory Inn Tolleson	460995068	The most disgusting motel rooms I have ever ex...
4	883723092	A Victory Inn Tolleson	431610872	After traveling for 2 day's from Oregon to Ari...

```
[ ] data = data.drop_duplicates(['review_id'],keep = 'first')
```

```
[ ] data = data[pd.notnull(data['name'])]
    data = data[pd.notnull(data['review_text'])]
```

```
[ ] data['name'] = data['name'] + ' ' + data['review_text']
    data = data[[col for col in data.columns if col != 'review_text']]
    data = data[[col for col in data.columns if col != 'review_id']]
    data.rename(columns={'name':'review_text'}, inplace=True)
    data.head()
```

	id	review_text
0	883723092	A Victory Inn Tolleson I was traveling between...
1	883723092	A Victory Inn Tolleson My car was broken into ...
2	883723092	A Victory Inn Tolleson I check in to hotel and...
3	883723092	A Victory Inn Tolleson The most disgusting mot...
4	883723092	A Victory Inn Tolleson After traveling for 2 d...

Figure 3.4.3.1 : Suppression des données supplémentaires

```
[ ] data = data.groupby('id')['review_text'].apply(lambda x: " ".join(x)).to_frame().reset_index()
    data.head(10)
```

	id	review_text
0	1043012029	Haven Hotel Had a tiny room overlooking the ba...
1	1169217642	Wulfrun Hotel i arrived on friday afternoon wi...
2	1297964848	Soldiers Sailors Marines Club Perhaps; years a...
3	1378230042	The Saint James Hotel; an Ascend Hotel Collect...
4	1382375475	Windemere Hotel and Conference Center Front de...
5	1404078972	Motel 6 Evansville dirty; not even a delivery ...
6	1422292673	So Paddington I bid for this room on Price lin...
7	1500055888	Four Points by Sheraton New York Downtown The ...
8	1583937125	Hotel Carter I know this hotel has had lots of...
9	1647557143	The Beverley Hotel The Facts: The hotel is sma...

Figure 3.4.3.2 : Exécution de suppression des données supplémentaires

4) Prétraitement des données

```
[ ]  def lemmatize(text):
         wordnet_lemmatizer = WordNetLemmatizer()
         return wordnet_lemmatizer.lemmatize(text)

     def preprocess(text):
         result=[]
         for token in gensim.utils.simple_preprocess(text) :
             if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
                 result.append(lemmatize(token))

         return result
```

```
[ ]  data['review_text'] = data['review_text'].apply(preprocess)
```

Figure 3.4.4 : Prétraitement des données

5) Déterminer le nombre de sujets

```
[ ] reviews = data['review_text']
    dictionary = gensim.corpora.Dictionary(reviews)
    bow_corpus = [dictionary.doc2bow(doc) for doc in reviews]
```

```
[ ] def compute_coherence_values(dictionary, corpus, texts, limit, start=2, step=3):
        coherence_values = []
        model_list = []
        for num_topics in range(start, limit, step):
            model=LdaModel(corpus=corpus, id2word=dictionary, num_topics=num_topics)
            model_list.append(model)
            coherencemodel = CoherenceModel(model=model, texts=texts, dictionary=dictionary, coherence='c_v')
            coherence_values.append(coherencemodel.get_coherence())

        return model_list, coherence_values
```

```
[ ] model_list, coherence_values = compute_coherence_values(dictionary=dictionary,
                                                            corpus=bow_corpus, texts=reviews, start=10, limit=20, step=1)
```

```
limit=20; start=10; step=1;
x = range(start, limit, step)

plt.plot(x,coherence_values)
plt.xlabel("Num Topics")
plt.ylabel("Coherence score")
plt.legend(("coherence_values"), loc='best')
plt.show()
```

Figure 3.4.5 : Déterminer le nombre de sujets

6) Construire un modèle LDA

```
[ ] lda_model =  gensim.models.LdaMulticore(bow_corpus,
                                            num_topics = 16,
                                            id2word = dictionary,
                                            passes = 10,
                                            workers = 2)
```

Figure 3.4.5 : Construire un modèle LDA

7) Répartition des sujets sur les mots (16 sujets, 10 mots)

```
[ ] for idx in range(16):
        print("Topic #%s:" % idx, lda_model.print_topic(idx, 10))

Topic #0: 0.030*"best" + 0.027*"hotel" + 0.027*"western" + 0.024*"room" + 0.014*"waterfront" + 0.014*"chateau" + 0.013*"vaudreuil" + 0.013*"lake" + 0.012*"staff" + 0.012*"breakfast"
Topic #1: 0.048*"apartment" + 0.032*"yourstay" + 0.022*"midtown" + 0.005*"herald" + 0.004*"key" + 0.004*"cable" + 0.004*"macy" + 0.003*"player" + 0.003*"room" + 0.003*"broadway"
Topic #2: 0.044*"resort" + 0.029*"forest" + 0.028*"gokarna" + 0.016*"staff" + 0.015*"room" + 0.015*"hotel" + 0.012*"good" + 0.012*"food" + 0.010*"golf" + 0.009*"stay"
Topic #3: 0.002*"room" + 0.001*"hotel" + 0.001*"stay" + 0.001*"nice" + 0.001*"breakfast" + 0.001*"great" + 0.001*"staff" + 0.001*"clean" + 0.000*"good" + 0.000*"place"
Topic #4: 0.035*"hotel" + 0.027*"room" + 0.014*"staff" + 0.012*"house" + 0.011*"great" + 0.011*"toronto" + 0.011*"downtown" + 0.010*"edinburgh" + 0.010*"good" + 0.009*"norton"
Topic #5: 0.080*"wyndham" + 0.068*"microtel" + 0.062*"kannapolis" + 0.062*"concord" + 0.044*"suite" + 0.018*"room" + 0.011*"towel" + 0.008*"staff" + 0.007*"desk" + 0.007*"ant"
Topic #6: 0.055*"hotel" + 0.040*"room" + 0.012*"night" + 0.009*"stay" + 0.009*"good" + 0.008*"bathroom" + 0.008*"staff" + 0.008*"place" + 0.008*"clean" + 0.008*"small"
Topic #7: 0.027*"room" + 0.017*"hotel" + 0.014*"breakfast" + 0.013*"staff" + 0.013*"stay" + 0.013*"great" + 0.012*"edinburgh" + 0.012*"residence" + 0.011*"good" + 0.011*"location"
Topic #8: 0.031*"hotel" + 0.029*"room" + 0.016*"kathmandu" + 0.015*"guest" + 0.013*"staff" + 0.013*"house" + 0.012*"stay" + 0.011*"good" + 0.010*"great" + 0.009*"place"
Topic #9: 0.008*"hotel" + 0.005*"room" + 0.002*"stay" + 0.002*"place" + 0.002*"staff" + 0.002*"good" + 0.002*"night" + 0.001*"clean" + 0.001*"breakfast" + 0.001*"friendly"
Topic #10: 0.036*"hotel" + 0.028*"room" + 0.011*"staff" + 0.010*"stay" + 0.010*"good" + 0.010*"place" + 0.009*"clean" + 0.009*"great" + 0.009*"collection" + 0.009*"nice"
Topic #11: 0.058*"airport" + 0.037*"hotel" + 0.020*"centaur" + 0.020*"room" + 0.013*"flight" + 0.012*"india" + 0.011*"stay" + 0.011*"courtyard" + 0.008*"margot" + 0.008*"independence"
Topic #12: 0.031*"hotel" + 0.031*"room" + 0.013*"night" + 0.011*"good" + 0.010*"stay" + 0.010*"staff" + 0.010*"place" + 0.009*"breakfast" + 0.008*"clean" + 0.006*"location"
Topic #13: 0.037*"room" + 0.032*"hotel" + 0.012*"stay" + 0.010*"night" + 0.009*"place" + 0.008*"clean" + 0.008*"suite" + 0.007*"staff" + 0.006*"good" + 0.006*"like"
Topic #14: 0.033*"hotel" + 0.028*"room" + 0.018*"house" + 0.017*"glidden" + 0.014*"breakfast" + 0.014*"tavern" + 0.012*"stay" + 0.011*"great" + 0.008*"staff" + 0.008*"restaurant"
Topic #15: 0.002*"hotel" + 0.002*"room" + 0.001*"stay" + 0.001*"staff" + 0.001*"night" + 0.001*"good" + 0.001*"breakfast" + 0.001*"place" + 0.001*"stayed" + 0.001*"time"
```

```
word_dict = {};
for i in range(16):
    words = lda_model.show_topic(i, topn = 10)
    word_dict['Topic # ' + '{:02d}'.format(i)] = [i[0] for i in words]
pd.DataFrame(word_dict)
```

	Topic # 00	Topic # 01	Topic # 02	Topic # 03	Topic # 04	Topic # 05	Topic # 06	Topic # 07	Topic # 08	Topic # 09	Topic # 10	Topic # 11	Topic # 12	Topic # 13	Topic # 14	Topic # 15
0	best	apartment	resort	room	hotel	wyndham	hotel	room	hotel	hotel	hotel	airport	hotel	room	hotel	hotel
1	hotel	yourstay	forest	hotel	room	microtel	room	hotel	room	room	room	hotel	room	hotel	room	room
2	western	midtown	gokarna	stay	staff	kannapolis	night	breakfast	kathmandu	stay	staff	centaur	night	stay	house	stay
3	room	herald	staff	nice	house	concord	stay	staff	guest	place	stay	room	good	night	glidden	staff
4	waterfront	key	room	breakfast	great	suite	good	stay	staff	staff	good	flight	stay	place	breakfast	night
5	chateau	cable	hotel	great	toronto	room	bathroom	great	house	good	place	india	staff	clean	tavern	good
6	vaudreuil	macy	good	staff	downtown	towel	staff	edinburgh	stay	night	clean	stay	place	suite	stay	breakfast
7	lake	player	food	clean	edinburgh	staff	place	residence	good	clean	great	courtyard	breakfast	staff	great	place
8	staff	room	golf	good	good	desk	clean	good	great	breakfast	collection	margot	clean	good	staff	stayed
9	breakfast	broadway	stay	place	norton	ant	small	location	place	friendly	nice	independence	location	like	restaurant	time

Figure 3.4.7 : Répartition des sujets sur les mots

8) Trouver le thème dominant pour chaque hôtel

```
[ ] def format_topics_sentences(ldamodel, corpus, texts):
        sent_topics_df = pd.DataFrame()

        for i, row in enumerate(ldamodel[corpus]):
            row = sorted(row, key=lambda x: (x[1]), reverse=True)
            for j, (topic_num, prop_topic) in enumerate(row):
                if j == 0:
                    wp = ldamodel.show_topic(topic_num)
    #                topic_keywords = ", ".join([word for word, prop in wp])
                    sent_topics_df = sent_topics_df.append(pd.Series([int(topic_num), round(prop_topic,4)]), ignore_index=True)
                else:
                    break
        sent_topics_df.columns = ['Dominant_Topic', 'Perc_Contribution']

        contents = pd.Series(texts)
        sent_topics_df = pd.concat([sent_topics_df, contents], axis=1)
        return(sent_topics_df)
```

```
[ ] df_topic_sents_keywords = format_topics_sentences(ldamodel=lda_model, corpus=bow_corpus, texts=reviews)

    df_dominant_topic = df_topic_sents_keywords.reset_index()
    df_dominant_topic.columns = ['Document_No', 'Dominant_Topic', 'Topic_Perc_Contrib', 'review_text']

    df_dominant_topic.head(10)
```

Figure 3.4.8.1 : Trouver le thème dominant pour chaque hotel

	Document_No	Dominant_Topic	Topic_Perc_Contrib	review_text
0	0	12.0	0.6110	[haven, hotel, tiny, room, overlooking, yard, ...
1	1	6.0	0.9967	[wulfrun, hotel, arrived, friday, afternoon, b...
2	2	8.0	0.5389	[soldier, sailor, marine, club, year, nice, pl...
3	3	10.0	0.9765	[saint, james, hotel, ascend, hotel, collectio...
4	4	13.0	0.9996	[windemere, hotel, conference, center, desk, s...
5	5	13.0	0.9997	[motel, evansville, dirty, delivery, menu, piz...
6	6	12.0	0.7464	[paddington, room, price, line, star, maybe, s...
7	7	10.0	0.7932	[point, sheraton, york, downtown, remarkable, ...
8	8	6.0	0.5359	[hotel, carter, know, hotel, lot, review, sist...
9	9	6.0	0.4088	[beverley, hotel, fact, hotel, small, room, st...

```
df_dominant_topic.shape
```

```
(103, 4)
```

```
[ ] df_dominant_topic['id'] = data['id']
    df_dominant_topic.head(10)
```

	Document_No	Dominant_Topic	Topic_Perc_Contrib	review_text	id
0	0	12.0	0.6110	[haven, hotel, tiny, room, overlooking, yard, ...	1043012029
1	1	6.0	0.9967	[wulfrun, hotel, arrived, friday, afternoon, b...	1169217642
2	2	8.0	0.5389	[soldier, sailor, marine, club, year, nice, pl...	1297964848
3	3	10.0	0.9765	[saint, james, hotel, ascend, hotel, collectio...	1378230042
4	4	13.0	0.9996	[windemere, hotel, conference, center, desk, s...	1382375475
5	5	13.0	0.9997	[motel, evansville, dirty, delivery, menu, piz...	1404078972
6	6	12.0	0.7464	[paddington, room, price, line, star, maybe, s...	1422292673
7	7	10.0	0.7932	[point, sheraton, york, downtown, remarkable, ...	1500055888
8	8	6.0	0.5359	[hotel, carter, know, hotel, lot, review, sist...	1583937125

Figure 3.4.8.2 : Exécution Trouver le thème dominant pour chaque hotel

9) répartition de l'hôtel sur les sujets (matrice document-sujet)

```
temp_list = []
for x in range(df_dominant_topic.shape[0]):
    new_doc = df_dominant_topic['review_text'][x]
    new_doc_bow = dictionary.doc2bow(new_doc)
    temp = lda_model.get_document_topics(new_doc_bow, minimum_probability=0.0)
    temp_list.append([round(n,5) for _, n in temp])

doc_topic_df_prob = pd.DataFrame(temp_list)

df_original_copy = df_dominant_topic

doc_topic_table = pd.concat([df_original_copy, doc_topic_df_prob], axis=1)

doc_topic_table = doc_topic_table[[col for col in doc_topic_table.columns if col not in
                                   ['Dominant_Topic','Topic_Perc_Contrib', 'review_text', 'id']]]
doc_topic_table.rename(columns={0: "Topic 0", 1: "Topic 1", 2: "Topic 2", 3: "Topic 3", 4: "Topic 4", 5: "Topic 5",
                                6: "Topic 6", 7: "Topic 7", 8: "Topic 8", 9: "Topic 9", 10: "Topic 10", 11: "Topic 11",
                                12: "Topic 12", 13: "Topic 13", 14: "Topic 14", 15: "Topic 15"},
                       inplace=True)
doc_topic_table.head(10)
```

	Document_No	Topic 0	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10	Topic 11	Topic 12	Topic 13	Topic 14	Topic 15
0	0	0.00001	0.00001	0.00001	0.00001	0.00059	0.00001	0.38017	0.00679	0.00001	0.00001	0.00001	0.00001	0.61085	0.00145	0.00001	0.00001
1	1	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003	0.99681	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003	0.00282	0.00003	0.00003
2	2	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.07786	0.11156	0.53893	0.00002	0.00103	0.00002	0.00002	0.27039	0.00002	0.00002
3	3	0.00000	0.00000	0.00000	0.00000	0.01086	0.00000	0.00031	0.00937	0.00000	0.00000	0.97592	0.00000	0.00000	0.00000	0.00350	0.00000
4	4	0.00007	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.99959	0.00025	0.00001
5	5	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.99971	0.00002	0.00002
6	6	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.24514	0.00001	0.00000	0.00000	0.00000	0.00000	0.74581	0.00899	0.00000	0.00000
7	7	0.00000	0.00000	0.00000	0.00000	0.02754	0.00000	0.13354	0.03970	0.00002	0.00000	0.79375	0.00000	0.00002	0.00539	0.00000	0.00000
8	8	0.00002	0.00002	0.00002	0.00002	0.00002	0.00002	0.53468	0.00085	0.00301	0.00002	0.43334	0.00002	0.00250	0.02547	0.00002	0.00002
9	9	0.00001	0.00001	0.00001	0.00001	0.19872	0.00001	0.40886	0.11183	0.00260	0.00001	0.01644	0.00001	0.00001	0.17503	0.08648	0.00001

Figure 3.4.9 : répartition de l' hotel sur les sujets

10) Trouver cinq pairs pour chaque hôtel avec une similitude cosinus)

```
[ ]  temp_list = []

     for x in range(df_dominant_topic.shape[0]):
         new_doc = df_dominant_topic['review_text'][x]
         new_doc_bow = dictionary.doc2bow(new_doc)
         temp = lda_model.get_document_topics(new_doc_bow, minimum_probability=0.0)
         temp_list.append([temp])

     df_sim = pd.DataFrame(temp_list)
```

```
[ ]  overall_temp_list = []
     for hotel in range(df_dominant_topic.shape[0]):
         temp_hotel_list = []
         for peer in range(df_dominant_topic.shape[0]):
             if peer != hotel:
                 first_hotel = df_sim.iloc[hotel][0]
                 second_hotel = df_sim.iloc[peer][0]

                 sim = gensim.matutils.cossim(first_hotel,second_hotel)
                 temp_hotel_list.append([peer,round(sim,5)])
         temp_hotel_list.sort(key = lambda x: x[1], reverse = True)
         overall_temp_list.append(temp_hotel_list[:5])
```

```
     temp_hotel_recommendation = pd.DataFrame(overall_temp_list)

     df_original_copy = df_dominant_topic

     recommendation_table = pd.concat([df_original_copy, temp_hotel_recommendation], axis=1)

     recommendation_table = recommendation_table[[col for col in recommendation_table.columns if col not in
                                          ['Document_No', 'Dominant_Topic','Topic_Perc_Contrib', 'review_text']]]

     recommendation_table.rename(columns={0: "First reco", 1: "Second reco", 2: "Third reco", 3: "Forth reco", 4: "Fifth reco"},
                                 inplace=True)

     recommendation_table.head(10)
```

Figure 3.4.10.1 : Trouver cinq pairs pour chaque hotel avec une similitude

	id	First reco	Second reco	Third reco	Forth reco	Fifth reco
0	1043012029	[97, 0.99964]	[26, 0.97757]	[6, 0.97116]	[33, 0.95037]	[45, 0.86904]
1	1169217642	[15, 1.0]	[40, 1.0]	[70, 1.0]	[72, 1.0]	[17, 0.99882]
2	1297964848	[88, 0.87252]	[73, 0.87234]	[54, 0.87225]	[79, 0.86956]	[31, 0.85147]
3	1378230042	[62, 0.99967]	[47, 0.99715]	[32, 0.99342]	[7, 0.98523]	[49, 0.97349]
4	1382375475	[5, 1.0]	[66, 1.0]	[68, 1.0]	[89, 1.0]	[102, 0.99996]
5	1404078972	[4, 1.0]	[66, 1.0]	[68, 1.0]	[89, 1.0]	[102, 0.99996]
6	1422292673	[26, 0.99958]	[0, 0.97116]	[97, 0.96495]	[45, 0.96166]	[76, 0.95031]
7	1500055888	[3, 0.98523]	[62, 0.98428]	[47, 0.98236]	[32, 0.97952]	[49, 0.95852]
8	1583937125	[94, 0.83958]	[84, 0.8181]	[40, 0.77641]	[1, 0.77636]	[70, 0.77628]
9	1647557143	[65, 0.88795]	[25, 0.88334]	[41, 0.87879]	[75, 0.86988]	[78, 0.86146]

Figure 3.4.10.2 : Exécution Trouver cinq pairs pour chaque hotel avec une similitude

5. Résultat et discussion

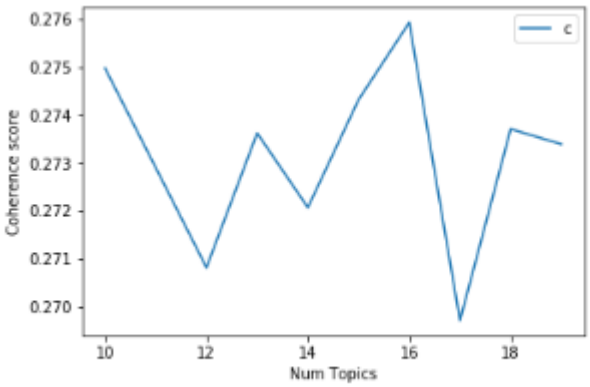

Figure 3.5 : la résultat d' Exécution

Le graphique est aussi proche de 0 qu'il devient parfait, la plupart des systèmes sont confinés à 0,5, comme nous remarquons que notre système est confiné à 0,275 et puisqu'il est plus proche de 0 que de 0,5, notre système est meilleur que la plupart des systèmes.

6. Conclusion

Dans cette section, nous avons présenté notre base de données, les outils de développement de ce système, et les étapes d'application. Nous avons également présenté les résultats de chaque étape de notre étude et les résultats d'évaluation de notre système. Les résultats de notre échantillon de données sont encourageants, mais une évaluation plus approfondie à l'aide d'autres mesures est encore nécessaire.

Conclusion générale

Les systèmes de recommandation ont reconnu une grande importance dans les différentes applications web. Ils sont utilisés pour aider les utilisateurs à sélectionner les objets et les produits qui correspondent le mieux à leurs besoins et préférences. L'objectif principal de notre travail, est de proposer un système de recommandation des hôtels où le calcul des prédictions se base sur l'application de la méthode LDA. Cette dernière est utilisée afin d'identifier les propriétés cachées entre les hôtels à partir des commentaires des clients. Le calcul du voisinage et des recommandations est donc effectué dans cet espace latent.

Comme perspectives de recherche nous envisageons d'intégrer d'autres types d'informations dans notre système ainsi que la validation du modèle proposé dans d'autres domaines d'application.

Références

A. Références Bibliographiques

1 Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Enquête sur les systèmes de recommandation. Systèmes basés sur la connaissance, 46, 109–132.

2 Resnick, P., & Varian, H.R. (1997). Systèmes de recommandation. Communication de l'ACM, 40, 56–58.

3 En ligneBurke, R. (2002). Systèmes hybrides de recommandation : enquête et expérimentations. Modélisation de l'utilisateur et interaction adaptée à l'utilisateur, 12(4), 331–370.

4 Belloui, A. (2008). L'usage des concepts du web sémantique dans le filtrage d'information collaboratif. Thèse de Doctorat, Ecole supérieur d'informatique, Alger.

5

6 Rao, N., & Talwar, V. (2008). Application domain and functional classification of recommender systems : a survey. Desidoc Journal of Library and Information Technology, 28(3), 17–36.

7 **(Blei et al., 2003)** Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. The Journal of Machine Learning Research, 3(null), 993–1022.

8 **(Yan et al., 2016)** Yan M, Sang J, Xu C, Hossain MS. A Unified Video Recommendation by Cross-Network User Modeling. ACM Trans Multimedia Comput Commun Appl. 2016;12(4):1–24.

9 **(Bagul and Barve)** D. V. Bagul and S. Barve, "A novel content-based recommendation approach based on LDA topic modeling for literature recommendation," 2021 6th International Conference on Inventive Computation Technologies (ICICT), 2021, pp. 954-961, doi:

10.1109/ICICT50816.2021.9358561.

10 Intelligence artificielle [cs.AI]. Université d'Avignon, 2017.

11 **(Herwanto and Ningtyas)** G. Herwanto and A. Ningtyas, "Recommendation system for web article based on association rules and topic modelling," Bulletin of Social Informatics Theory and Application, vol. 1, pp. 26–33, 03 2017.

B. Références Web (Techniques)

[W1] **(Jain et al., 2020)** Jain, G., Mahara, T., & Tripathi, K. N. (2020). A Survey of Similarity Measures for Collaborative Filtering-Based Recommender System. In M. Pant, T. K. Sharma, O. P. Verma, R. Singla, & A. Sikander (Eds.), Soft Computing: Theories and Applications (pp. 343–352). Springer. https://doi.org/10.1007/978-981-15-0751-9_32

[W2] https://fr.wikipedia.org/wiki/Syst%C3%A8me_de_recommandation/

[W3] https://www.wikiwand.com/fr/Syst%C3%A8me_de_recommandation

[W4] https://www.msn.com/fr-xl/

[W5] Al-Ghossein M, Murena P-A, Abdessalem T, Barré A, Cornuéjols A. Adaptive collaborative topic modeling for online recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems. British Columbia, Canada: Association for Computing Machinery; 2018. p. 338–346. (RecSys '18). https://doi.org/10.1145/3240323.3240363

[W6] https://www.ummto.dz/dspace/bitstream/handle/ummto/13103/Benkhouya%20B,%20Ait%20Abdelmalek%20R..pdf?sequence=1

[W7] https://archives.refad.ca/evaluation_en_ligne.pdf

C. Références figure

[***1.4.1***] https://interstices.info/wp-content/uploads/2018/06/RS-Categ-fig1-700.jpg

[***1.4.3***] https://interstices.info/wp-content/uploads/2018/06/RS-Categ-fig2-700.jp

[***1.4.3***] https://interstices.info/wp-content/uploads/2018/06/RS-Categ-fig5-700.jpg

Résumé

Le travail présenté dans ce manuscrit s'inscrit dans le domaine des systèmes de recommandation qui est devenu une méthodologie dominante dans la majorité des applications web. Les systèmes de recommandation sont des outils prometteurs afin de générer des services pertinents aux utilisateurs. L'objectif principal de notre travail est de proposer un algorithme de recommandation d'hôtel qui se base sur l'allocation de Dirichlet latente (LDA). L'algorithme proposé réduit l'espace de dimensionnalité en extrayant les propriétés latentes des textes non structuré représentant les avis des clients sur les hôtels. Ces propriétés sont modélisées sous forme de vecteurs numériques qui sont automatiquement appris grâce à l'application de l'algorithme LDA. Le calcul des similarités en se basant sur ses thèmes latents permet de découvrir les relations cachées entre les commentaires des utilisateurs pour faire des recommandations pertinentes.

Abstract

The work presented in this manuscript is part of the field of recommender systems which has become a dominant methodology in the majority of web applications. Recommender systems are promising tools for generating relevant services to users. The main objective of our work is to propose a hotel recommendation algorithm based on the latent Dirichlet allocation (LDA). The proposed algorithm reduces the dimensionality space by extracting latent properties from unstructured texts representing guest reviews of hotels. These properties are modeled as digital vectors which are automatically learned through the application of the LDA algorithm. The calculation of similarities based on its latent themes makes it possible to discover the hidden relationships between user comments to make relevant recommendations.

ملخص

العمل المقدم في هذه المخطوطة هو جزء من مجال أنظمة التوصية التي أصبحت منهجية سائدة في غالبية تطبيقات الويب.أنظمة التوصية هي أدوات واعدة لتوليد الخدمات ذات الصلة للمستخدمين. الهدف الرئيسي من عملنا هو اقتراح خوارزمية توصية فندقية بناءً على تخصيص Dirichlet الكامن (LDA). تقلل الخوارزمية المقترحة مساحة الأبعاد عن طريق استخراج الخصائص الكامنة من النصوص غير المهيكلة التي تمثل تقييمات النزلاء للفنادق. تم تصميم هذه الخصائص على شكل نواقل رقمية يتم تعلمها تلقائيًا من خلال تطبيق خوارزمية LDA. يتيح حساب أوجه التشابه بناءً على سماته الكامنة اكتشاف العلاقات المخفية بين تعليقات المستخدم لتقديم التوصيات ذات الصلة.