LOG6308 - TP 2

Grégoire Chapeaux - 2033122 — Hugo Canneddu - 2096673

21 03 2021

Ce TP porte sur la recommandation d'articles scientifiques, par le biais de différentes méthodes (basées sur les références, les relations entre articles, le contenu, etc). Le TP se base sur des données relatives à 1090 articles extraites de la base de données Citeseer.

1 Algorithme de recommandation Bayésien - 4 pts

Cette question a été reportée depuis le TP1. L'objectif était d'implémenter un système de recommandation par une approche Bayésienne, basée sur l'appartenance d'un utilisateur à différentes catégories : genre (masculin ou féminin), job (ingénieur, artiste, étudiant...) et groupe d'âge (moins de 25 ans ou 25 ans et plus).

En notant job_U, $gender_U$ et age_U les évidences d'appartenance aux différentes catégories de U, et I l'hypothèse "aimer l'item I", on calcule le ratio de chance à l'aide de la formule suivante :

$$O(I|job_U, gender_U, age_U) = \frac{P(I)}{P(\neg I)} \frac{P(job_U|I)}{P(job_U|\neg I)} \frac{P(gender_U|I)}{P(gender_U|\neg I)} \frac{P(age_U|I)}{P(age_U|\neg I)}$$

Dès lors, on calcule la probabilité pour U d'aimer le film I :

$$P(U \heartsuit I) = \frac{O(I|job_U, gender_U, age_U)}{1 + O(I|job_U, gender_U, age_U)}$$

. En calculant, pour un film donné, le vote moyen d'appréciation V_{like} (moyenne des votes strictement supérieurs à 3) et de non-appréciation $V_{dislike}$ (moyenne des votes inférieurs ou égaux à 3), on peut alors prédire le vote de U pour le film I :

$$V_{U,I} = P(U \heartsuit I) \cdot V_{like} + (1 - P(U \heartsuit I)) \cdot V_{dislike}$$

L'approche a été implémentée en Python, en raisonnant directement sur des DataFrames et en calculant les ratios de chance sur l'ensemble des films pour chaque utilisateur. Par convention, lorsqu'une observation est manquante, on applique une correction de Laplace en supposant avoir une observation de plus pour chaque cas, et lorsque l'information est manquante, on fixe le ratio de chance à 1 (si on ne sait rien de la situation, on a autant de chances d'aimer le film que de ne pas l'aimer).

Cette approche nous permet de calculer une prédiction sur tous les utilisateurs pour tous les films. Par simplicité, nous avons produit une seule prédiction sur tout l'ensemble pour vérifier l'erreur globale, mais une validation croisée pourrait donner une erreur plus représentative de la réalité. Finalement, en calculant l'erreur entre la prédiction et l'observation, on trouve une **MSE de 0.909**.

Par exemple, pour une femme de 23 ans ingénieure, les 10 films recommandés sont résumés dans la table suivante :

Requête : Femme, Jeune, Ingénieure		
Rang	Titre	Note prédite
1	Paradise Lost: The Child Murders at Robin Hood Hills (1996)	4.656
2	Casablanca (1942)	4.631
3	Wallace & Gromit: The Best of Aardman Animation (1996)	4.524
4	Schindler's List (1993)	4.515
5	Harlem (1993)	4.5
6	Maya Lin: A Strong Clear Vision (1994)	4.5
7	Walking and Talking (1996)	4.492
8	Star Wars (1977)	4.485
9	Turbulence (1997)	4.466
10	Princess Bride, The (1987)	4.460

Table 1: Recommandations par approche bayésienne

2 Pagerank - 4 pts

Dans un premier temps, nous avons réalisé le calcul du Pagerank sur l'ensemble des documents. Afin d'éviter des divisions par zéro dans l'algorithme de Pagerank, nous avons fixé la diagonale de la matrice d'adjacence à 1. Ainsi nous considérons que tous les documents se référencent eux-mêmes.
Une fois le Pagerank calculé pour tous les documents étudiés, nous avons dans un premier temps extrait pour le document 422908 les documents qui ont le meilleur Pagerank parmi les documents référencés par le document 422908. Dans un second temps, nous avons étendu cet espace en ajoutant les documents qui sont des références des références, pour ce faire nous avons pris la matrice d'adjacence à laquelle on ajouter la matrice d'adjacente au carré, qui représente les liens de deuxièmes degrés.
Dans les résultats obtenu on peut observer certaines anomalies (document avec un titre *null*) : il s'agit en réalité de document sur les quels les informations extraite de Citeseer semblent incomplètes. Ces documents n'ont donc pas de lien sortant mais uniquement des liens entrants issus d'autres documents. Dans l'implémentation utilisée de Pagerank les documents sans lien sortant ne vont pas redistribuer leur valeur de Pagerank, ils vont donc seulement "absorber" la valeur des documents qui les référencent, d'où une valeur élevé de Pagerank. On obtient ainsi les recommandations suivantes :

Requête : 442908 - Symbolic Model Checking for Real-time Systems			
Rang	Id	Titre	Pagerank
1	311874	Graph-Based Algorithms for Boolean Function Manipulation	0.07829
2	396568	The Existence of Refinement Mappings	0.01173
3	522428	Null	0.01052
4	19422	Symbolic Model Checking: 10 20 States and Beyond	0.00777
5	17094	A Really Temporal Logic	0.00338
6	155792	Logics and Models of Real Time: A Survey	0.00235
7	64835	The Benefits of Relaxing Punctuality	0.00234
8	315693	An Old-Fashioned Recipe for Real Time	0.00216
9	110303	Temporal Proof Methodologies for Real-time Systems	0.00185
10	241538	Deciding Properties Of Regular Real Timed Processes	0.00162

Table 2: Recommandations par Pagerank classique

Requête : 442908 - Symbolic Model Checking for Real-time Systems			
Rang	Id	Titre	Pagerank
1	311874	Graph-Based Algorithms for Boolean Function Manipulation	0.07829
2	396568	The Existence of Refinement Mappings	0.01173
3	522428	Null	0.01052
4	389559	A Fast Mutual Exclusion Algorithm	0.00808
5	19422	Symbolic Model Checking: 10 20 States and Beyond	0.00777
6	225173	Symbolic Model Checking with Partitioned Transition Relations	0.00344
7	425638	Delay Analysis in Synchronous Programs	0.00343
8	17094	A Really Temporal Logic	0.00338
9	70445	Real-time Logics: Complexity and Expressiveness	0.00289
10	206738	A Partial Approach to Model Checking	0.00260

Table 3: Recommandations par Pagerank avec voisinage étendu : références et références des références

3 Cos-similarité - 2 pts

En utilisant la similarité Cosinus, on peut calculer la similarité entre deux articles. De là, on peut facilement calculer les 10 articles les plus similaires à l'article 442908 :

Requête : 442908 - Symbolic Model Checking for Real-time Systems		
Rang	Id	Titre
1	96767	Model-Checking in Dense Real-time
2	70445	Real-time Logics: Complexity and Expressiveness
3	466838	Sooner is Safer than Later
4	149673	Back to the Future: Towards a Theory of Timed Regular Languages
5	53632	A Theory of Timed Automata
6	155792	Logics and Models of Real Time: A Survey
7	497542	Compiling Real-Time Specifications into Extended Automata
8	3175	Parametric Real-time Reasoning
9	17507	The Algorithmic Analysis of Hybrid Systems
10	147460	Minimization of Timed Transition Systems

Table 4: Recommandations par Cos-similarité

4 Recommandations par approche basée sur le contenu - 2 pts

Cette méthode se base sur une approche contenu pour trouver les documents qui sont les plus similaires à un document donné. Nous avons cherché à calculer la valeur de TF-IDF entre tous les documents étudiés. Pour ce faire nous avons dans un premier temps essayé d'appliquer nous-mêmes un certain nombre de transformations dont :

- L'élimination des caractères spéciaux, chiffre et ponctuation
- La tokenisation
- L'élimination du vocabulaire des termes les plus fréquents
- L'élimination des stop words
- L'application d'une racinisation ou d'une lemmatisation

Cependant en comparant nos résultats avec prétraitement et sans prétraitement en utilisant TfidfVectorizer de sklearn, nous avons observé de moins bon résultats lorsque nous appliquons nos prétraitements, cela étant certainement dû au fait que TfidfVectorizer a déjà un certain nombre de prétraitements qu'il peut

appliquer de façon native. De ce fait, nous avons utilisé TfidfVectorizer sur notre corpus avec comme seul prétraitement une racinisation, n'étant pas implémenté de base dans l'algorithme de sklearn, contrairement aux autres transformations que nous avions appliqué.
Les résultats obtenus pour cette approche sont rassemblés dans la table suivante :

Requête : 442908 - Symbolic Model Checking for Real-time Systems		
Rang	Id	Titre
1	373307	An Efficient Generation of the Timed Reachability Graph for the Analysis of Real-Time Systems
2	426325	What Good Are Digital Clocks?
3	53595	Reducing the Number of Clock Variables of Timed Automata
4	147460	Minimization of Timed Transition Systems
5	120172	Hardware Timing Verification using KRONOS
6	110303	Temporal Proof Methodologies for Real-time Systems
7	96767	Model-Checking in Dense Real-time
8	322240	Verification of Linear Hybrid Systems By Means of Convex Approximations
9	149673	Back to the Future: Towards a Theory of Timed Regular Languages
10	155792	Logics and Models of Real Time: A Survey

Table 5: Recommandations par approche basée sur le contenu

5 Recommandation d'article - 8 pts

Dans cette partie nous avons cherché à générer des recommandations d'article en combinant les différentes approches précédemment utilisées.

Pour ce faire, nous avons dans un premier temps généré 20 recommandations pour chaque articles grâce à l'approche contenu et le score de TF-IDF basé sur la description des articles. Contrairement à la question 4, nous n'avons pas fait manuellement de pré-traitement de racinisation, nous avons simplement utilisé ceux déjà présent dans *TfidfVectorizer* car nous avons remarqué que c'est ce qui nous avait donné les meilleurs résultats.
Dans un second temps, nous avons généré 20 recommandations en nous basant sur l'approche cosinus de la même façon qu'à la question 3.

Une fois ces deux listes de recommandation obtenues, nous les avons combiné en les sommant, ce qui nous permet de savoir si un article est recommandé par aucune, l'une ou les deux approches.

Finalement, pour déterminer les 20 articles à retenir, nous avons calculé le Pagerank de chaque article que nous avons multiplié avec la matrice de nos résultats de recommandation précédente. De ce fait les articles qui sont recommandés 2 fois voient leurs score de Pagerank amélioré. Enfin nous avons simplement récupéré les 20 meilleurs scores pour déterminer nos recommandations finales.
Nous obtenons ainsi comme taux de rappel : **34.97%**.